H 6

```
TTTTTTTTTTTTTTT  TTTTTTTTTTTTTTT  DDDDDDDDDDDD     RRRRRRRRRRRR     VVV         VVV  RRRRRRRRRRRR
TTTTTTTTTTTTTTT  TTTTTTTTTTTTTTT  DDDDDDDDDDDD     RRRRRRRRRRRR     VVV         VVV  RRRRRRRRRRRR
TTTTTTTTTTTTTTT  TTTTTTTTTTTTTTT  DDDDDDDDDDDD     RRRRRRRRRRRR     VVV         VVV  RRRRRRRRRRRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRR         RRR  VVV         VVV  RRR         RRR
      TTT              TTT        DDD         DDD  RRRRRRRRRRRR     VVV         VVV  RRRRRRRRRRRR
      TTT              TTT        DDD         DDD  RRRRRRRRRRRR     VVV         VVV  RRRRRRRRRRRR
      TTT              TTT        DDD         DDD  RRRRRRRRRRRR     VVV         VVV  RRRRRRRRRRRR
      TTT              TTT        DDD         DDD  RRR   RRR        VVV         VVV  RRR   RRR
      TTT              TTT        DDD         DDD  RRR   RRR        VVV         VVV  RRR   RRR
      TTT              TTT        DDD         DDD  RRR   RRR        VVV         VVV  RRR   RRR
      TTT              TTT        DDD         DDD  RRR      RRR        VVV   VVV     RRR      RRR
      TTT              TTT        DDD         DDD  RRR      RRR        VVV   VVV     RRR      RRR
      TTT              TTT        DDD         DDD  RRR      RRR        VVV   VVV     RRR      RRR
      TTT              TTT        DDDDDDDDDDDD     RRR         RRR        VVV        RRR         RRR
      TTT              TTT        DDDDDDDDDDDD     RRR         RRR        VVV        RRR         RRR
      TTT              TTT        DDDDDDDDDDDD     RRR         RRR        VVV        RRR         RRR
```

```
_$


Va
--
00
00
00
00
00
00
00
48
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
80
```

```
**FILE**ID**TTYSTRSTP                                              K  5
```

```
TTTTTTTTTT  TTTTTTTTTT  YY      YY    SSSSSSSS  TTTTTTTTTT  RRRRRRRR      SSSSSSSS  TTTTTTTTTT  PPPPPPPP
TTTTTTTTTT  TTTTTTTTTT  YY      YY    SSSSSSSS  TTTTTTTTTT  RRRRRRRR      SSSSSSSS  TTTTTTTTTT  PPPPPPPP
    TT          TT      YY      YY  SS              TT      RR      RR  SS              TT      PP      PP
    TT          TT      YY      YY  SS              TT      RR      RR  SS              TT      PP      PP
    TT          TT        YY  YY    SS              TT      RR      RR  SS              TT      PP      PP
    TT          TT        YY  YY    SS              TT      RR      RR  SS              TT      PP      PP
    TT          TT          YY        SSSSSS    TT      RRRRRRRR      SSSSSS    TT      PPPPPPPP
    TT          TT          YY        SSSSSS    TT      RRRRRRRR      SSSSSS    TT      PPPPPPPP
    TT          TT          YY              SS      TT      RR  RR          SS      TT      PP
    TT          TT          YY              SS      TT      RR  RR          SS      TT      PP
    TT          TT          YY              SS      TT      RR    RR        SS      TT      PP          ....
    TT          TT          YY              SS      TT      RR    RR        SS      TT      PP          ....
    TT          TT          YY      SSSSSSSS    TT      RR      RR  SSSSSSSS    TT      PP          ....
    TT          TT          YY      SSSSSSSS    TT      RR      RR  SSSSSSSS    TT      PP          ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

TT
V0

```
0000            .TITLE  TTYSTRSTP - Terminal driver start/stop I/O routines
0000            .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000
0000    ;++
0000    ; FACILITY:
0000    ;
0000    ;       VAX/VMS TERMINAL DRIVER
0000    ;
0000    ; ABSTRACT:
0000    ;
0000    ;       THIS MODULE CONTAINS ROUTINES PERTAINING TO STARTING AND COMPLETING
0000    ;       I/O REQUESTS.
0000    ;
0000    ; AUTHOR:
0000    ;
0000    ;       R.HEINEN 10-OCT-1977
0000    ;
0000    ; Revision history:
0000    ;
0000    ;       V03-030 MIR0450         MICHAEL I. ROSENBLUM            27-JUN-1984
0000    ;               Add code to the free linefeed logic to account for PC_NOCRLF.
0000    ;               Fix problem that causes the first linefeed typed on a
0000    ;               read with no prompt to not be echoed.
0000    ;
0000    ;       V03-029 RKS0029         RICK SPITZ                      10-APR-1984
0000    ;               Enhance virtual terminal connect action routine to
0000    ;               perform an implicit set mode operation.
0000    ;
0000    ;       V03-028 MIR0370         Michael I. Rosenblum            20-Mar-1984
0000    ;               Put code in to fix problems with lines and prompts causing
0000    ;               wrap.  Fix bug that would cause FMS programs to crash the
0000    ;               system.
0000    ;
```

```
0000   ;       V03-027 RKS0027         RICK SPITZ                      05-MAR-1984
0000   ;               Enhance write post completion to handle the case
0000   ;               of a write completion with no current PUCB.
0000   ;
0000   ;       V03-026 MIR0310         Michael I. Rosenblum            09-Feb-1984
0000   ;               Fix bugs.
0000   ;               make sure setting nomodem on a modem terminal shuts down
0000   ;                   the line.
0000   ;
0000   ;       V03-025 MIR0300         Michael I. Rosenblum            30-Jan-1984
0000   ;               Add input fallback
0000   ;
0000   ;       V03-024 MIR0085         Michael I. Rosenblum            26-Aug-1983
0000   ;               Remove references to DCL_OUTBND and DCL_CTRLC.
0000   ;
0000   ;       V03-023 MIR0082         Michael I. Rosenblum            19-Aug-1983
0000   ;               Make autoxoff mode work with passall and ttsync.
0000   ;               Fix pasthru to remain enabled after a read completes.
0000   ;
0000   ;       V03-022 MIR0080         Michael I. Rosenblum            28-Jul-1983
0000   ;               Move newline code into write done rather than TTYFDT
0000   ;               Reposition routines in the module.
0000   ;
0000   ;       V03-021 MIR0070         Michael I. Rosenblum            13-jul-1983
0000   ;               Fix bug that would cause TTY$DISCONNECT to be called twice.
0000   ;               if a SETMODE with the HANGUP modifier was issued.
0000   ;
0000   ;       V03-020 MIR0051         Michael I. Rosenblum            23-Jun-1983
0000   ;               Fix missing litterals in connect and disconnect code.
0000   ;               Check write active bit in getnextwrite to insure that
0000   ;               The write queue is not reordered.
0000   ;
0000   ;       V03-019 RKS0019         RICK SPITZ                      7-JUN-1983
0000   ;               ADD CONNECT/DISCONNECT ACTION ROUTINES.
0000   ;               ENHANCE WRITE DONE FORK PROCESS TO ALWAYS USE REQCOM
0000   ;               IF THE WRITE IRP IS POINTED TO BY UCB$L_IRP
0000   ;               MAKE SURE LUCB IS NOT DETACHED AT THE ALTERNATE WRITE ENTRY.
0000   ;               REMOVE CTRLY HANGUP CHECK, AS IT IS STILL DONE IN FDT.
0000   ;
0000   ;       V03-018 RKS0018         RICK SPITZ                      16-MAY-1983
0000   ;               MOVE SEGMENTS OF CHARACTERISTICS FDT CODE TO TTYSTRSTP
0000   ;               TO ALLOW CLEAN DISCONNECT OF DISCONNECTED TERMINALS.
0000   ;               RESTORE LUCB FROM LUCB IN READ/WRITE DONE.
0000   ;
0000   ;       V03-017 MIR0050         Michael I. Rosenblum            11-May-1983
0000   ;               Remove code that special cased broadcasts.  Allow the
0000   ;               data returned by timeout errors to be stored in the recall
0000   ;               buffer.  Make write post complete broadcasts.
0000   ;
0000   ;       V03-016 MIR0030         Michael I. Rosenblum            30-Mar-1983
0000   ;               Integrate Read verification with the standard driver
0000   ;               Add support for alternate frame sizes.
0000   ;
0000   ;       V03-015 MIR0029         Michael I. Rosenblum            22-Mar-1983
0000   ;               Add field to the iosb when itemlist reads are used.
0000   ;
0000   ;       V03-014 RKS0014         RICK SPITZ                      14-MAR-1983
```

```
0000    ;               ADD SUPPORT FOR LOGICAL UCB. NOTE THAT THE DRIVER
0000    ;               SWITCHES TO PHYSICAL UCB CONTEXT AT STARTIO ENTRY. IT
0000    ;               RESTORES LOGICAL UCB CONTEXT PRIOR TO RETURNING TO THE
0000    ;               SYSTEM.
0000    ;
0000    ;       V03-013 MIR8026         Michael I. Rosenblum            14-Mar-1983
0000    ;               Fix bug in partail escape sequence processing.
0000    ;
0000    ;       V03-012 MIR5026         Michael I. Rosenblum            10-Mar-1983
0000    ;               Fix security whole with command recall and the password
0000    ;               by not allowing noecho strings to be stored in the recall
0000    ;               buffer.
0000    ;
0000    ;       V03-011 MIR1024         Michael I. Rosenblum            09-Mar-1983
0000    ;               Fix code in getnxtwrite to look at the read packet
0000    ;               rather than UCB$W_BCNT to find the number of characters
0000    ;               that have been read so far.
0000    ;
0000    ;       V03-010 MIR0026         Michael I. Rosenblum            01-Mar-1983
0000    ;               Add code to save the results of the last read.
0000    ;
0000    ;       V03-009 MIR0024         Michael I. Rosenblum            28-Jan-1983
0000    ;               Update code to use the new read packet format
0000    ;
0000    ;       V03-008 MIR0023         Michael I. Rosenblum            24-Jan-1983
0000    ;               Read buffer was used after it was deallocated if a
0000    ;               Cancel was issued while EDITREAD state was in affect.
0000    ;               Changed READONE code to clear the edit read states
0000    ;               when a read is completed.
0000    ;
0000    ;       V03-007 MIR0016         Michael I. Rosenblum            4-Jan-1983
0000    ;               Change TTY$STARTOUTPUT to use the UCB OUTYPE field to determine
0000    ;               the necessary action when TTY$GETNEXTCHAR is called.  This change
0000    ;               should illiminate the checking the volitale condition code bits
0000    ;               that previously had the function of OUTYPE.  For compatibility
0000    ;               purposes only we are setting the correct condition codes.
0000    ;
0000    ;       V03-006 MIR0015         Michael I. Rosenblum            20-Dec-1982
0000    ;               Change TTY$V_ST_UNSOL and TTY$V_ST_GETAHD to TTY$V_FD_UNSOL
0000    ;               and TTY$V_FD_GETAHD, to reflect changes in the fork dispatcher
0000    ;               also change PORT_DISCONNECT call to refer to
0000    ;               CLASS_MODEM_DIS.  Change all port calls to call the Class
0000    ;               driver jacket routines.
0000    ;
0000    ;       V03-005 MIR0013         Michael I. Rosenblum            16-Dec-1982
0000    ;               Fix up refferences to new ucb structure
0000    ;
0000    ;       V03-004 MIR0011         Michael I. Rosenblum            18-Nov-1982
0000    ;               Remove code that implimented HOLDSCREEN.
0000    ;
0000    ;       V03-003 MIR0010         Michael I Rosenblum     09-Nov-1982
0000    ;               Move the address of the terminator mask, and the length
0000    ;               of the prompt string from the IRP into the terminal read
0000    ;               buffer.
0000    ;
0000    ;       V03-002 ROW0077         Ralph O. Weber          27-MAR-1982
0000    ;               Change TTY$WRITEDONE to insure that eventhough UCB$W_TT_CURSOR
```

```
0000   ;                can now be bigger than UCB$W_DEVBUFSIZ, i.e. eventhough our
0000   ;                internal cursor position marker can virtually be beyond the
0000   ;                right-hand edge of the screen, the cursor-position value
0000   ;                returned in IOSB will never exceed the width of the screen.
0000   ;
0000   ;        V03-001 JLV0202         Jake VanNoy             23-MAR-1982
0000   ;                Change MODHANGUP from NOMOD to PRIV_TO_MOD in Set
0000   ;                Mode/Char logic.
0000   ;                Correct alternate class name lookup.
0000   ;
0000   ;        V02-045 RKS0045         RICK SPITZ              22-FEB-1982
0000   ;                Repair diagnostic function code logic.
0000   ;
0000   ;        V02-044 RKS0044         RICK SPITZ              16-FEB-1982
0000   ;                Enhance broadcast logic to allow delay prior to
0000   ;                forcing output. Move setting of controls pending
0000   ;                to STOP2 timeout. This way user ^s can be distinquished
0000   ;                from terminal xoff.
0000   ;                Save R3 prior to forking to create typeahead on read.
0000   ;
0000   ;        V02-043 RKS0043         RICK SPITZ              11-FEB-1982
0000   ;                Zero fork byte in TWP to allow DMA of broadcast.
0000   ;                Prevent XON characteristic from being permantly set.
0000   ;
0000   ;        V02-042 RKS0042         Rick Spitz              8-FEB-1982
0000   ;                Repair Alternate typeahead logic to allow setting
0000   ;                Permanent from users terminal.
0000   ;                Allocate typeahead buffer when starting read, if not already
0000   ;                done. This is needed for lines which are used for communications
0000   ;                on DMF-32 async lines.
0000   ;
0000   ;        V02-041 ROW0066         Ralph O. Weber          31-JAN-1982
0000   ;                Enhance alternate class driver setup to relocate address in
0000   ;                alternate class driver vector table.  Correct use of
0000   ;                unrelocateable .ASCID directive.
0000   ;
0000   ;        V02-040 RKS0040         RICK SPITZ              24-JAN-1982
0000   ;                USE INPUT VALUE FOR READ FIELD OFFSET.
0000   ;                ADD LOGIC TO BIND TO ALTERNATE DRIVER.
0000   ;
0000   ;        V02-039 RKS0039         RICK SPITZ              15-DEC-1981
0000   ;               FIX MAINTENANCE DISPATCH LOGIC.
0000   ;                DISALLOW SETTING ALT TYPEAHEAD IF ONE ALREAY EXISTS.
0000   ;                REMOVE LOGIO REQUIREMENT FOR PARITY ENABLE.
0000   ;                FIX WRTSTARTIO RETURN ADDRESSING.
0000   ;                ALLOW NOECHO READ TO NOT BLOCK WRITES.
0000   ;                ADD WRITE POST ROUTINE TO REPLACE INSPOST LOGIC, THIS
0000   ;                CORRECTS RACE CONDITION IN HALF DUPLEX WRITE COMPLETIONS.
0000   ;                ADD SUPPORT FOR ALTERNATE CLASS DRIVER.
0000   ;
0000   ;        V02-038 JLV0126         Jake VanNoy             1-Dec-1981
0000   ;                Add local echo logic and set speed privilege checking.
0000   ;
0000   ;        V02-037 JLV0102         Jake VanNoy             27-Oct-1981
0000   ;                Changed TTYDEFS to $TTYDEFS.
0000   ;
0000   ;        V02-036 JLV0070         Jake VanNoy                     28-Aug-1981
```

```
0000   229 ;                   Added UCB$L_TT_DEVDP1 checking and no refresh on broadcast.
0000   230 ;
0000   231 ;          V02-035 RKS035              RICK SPITZ                    26-AUG-1981
0000   232 ;                   ADD MAINT ENABLE BIT
0000   233 ;
0000   234 ;          V02-034 RKS034              RICK SPITZ                    20-AUG-1981
0000   235 ;                   ADD SUPPORT FOR ESCAPE MODIFIER ON READ.
0000   236 ;
0000   237 ;          V02-033 RKS033              RICK SPITZ                    12-AUG-1981
0000   238 ;                   RESET DMA ABORT STATE IN WRITE DONE LOGIC.
0000   239 ;                   RESET CONTROLS STATE FOR MAINTENANCE FUNCTIONS.
0000   240 ;
0000   241 ;          V02-032 RKS032              RICK SPITZ                    27-JULY-1981
0000   242 ;                   SEVERAL ENHANCEMENTS HAVE BEEN ADDED TO SUPPORT REVISIONS
0000   243 ;                   TO THE UCB STRUCTURE INCLUDING SPLIT SPEED, AND QUADWORD STATE
0000   244 ;                   AND DEVDEPEND FIELDS.
0000   245 ;                   SUPPORT FOR DIAGNOSTIC FUNCTIONS AND ENHANCED MODEM PROCESSING
0000   246 ;                   HAS BEEN ADDED. SEVERAL CHANGES TO SUPPORT THE CLASS/PORT
0000   247 ;                   STRUCTURE AS WELL AS ENHANCEMENTS TO ALLOW TERMINAL
0000   248 ;                   INITIATED CONTROL S AND Q DURING BROADCAST HAVE ALSO BEEN
0000   249 ;                   ADDED.
0000   250 ;
0000   251 ;          V02-031 RKS031              RICK SPITZ                    26-FEB-1981
0000   252 ;                   REMOVE V2.0 AUDIT TRAILS
0000   253 ;
```

```
    0000               .SBTTL  Declarations
    0000
    0000       ;
    0000       ; EXTERNAL SYMBOLS
    0000       ;
    0000               $ARBDEF                         ; DEFINE ACCESS RIGHTS BLOCK
    0000       ;       $CADEF                          ; DEFINE CONDITIONAL ASSEMBLY PARAMETERS.
    0000               $CRBDEF                         ; DEFINE CRB
    0000               $DPTDEF                         ; DEFINE DPT OFFSETS
    0000               $IODEF                          ; DEFINE I/O FUNCTION CODES
    0000               $IPLDEF                         ; DEFINE IPL'S
    0000               $IRPDEF                         ; DEFINE IRP
    0000               $PRDEF                          ; DEFINE PROCESSOR REGISTERS
    0000               $PRVDEF                         ; DEFINE PRIVILEGE MASK BITS
    0000               $SSDEF                          ; Define system status codes
    0000               $TTDEF                          ; DEFINE TERMINAL CHARACTERISTICS
    0000               $TT2DEF                         ; DEFINE TERMINAL CHARACTERISTICS
    0000               $TTYDEF                         ; DEFINE TERMINAL DRIVER SYMBOLS
    0000               $UCBDEF                         ; DEFINE UCB
    0000               $VECDEF                         ; DEFINE CRB VECTOR OFFSETS
    0000               $TTYMACS                        ; DEFINE TERMINAL MACROS
    0000               $TTYDEFS                        ; DEFINE TERMINAL DEFINITIONS
    0000               $TTYMODEM                       ; DEFINE TERMINAL MODEM DEFINITIONS
    0000
00000000               .PSECT  $$$115_DRIVER,LONG      ; DEFINE NON-PAGED PSECT
```

```
                              0000          .SBTTL  TTY$STARTIO - START I/O OPERATION ON TERMINAL
                              0000     ;++
                              0000     ; TTY$STARTIO - START I/O OPERATION ON TERMINAL
                              0000     ;
                              0000     ; FUNCTIONAL DESCRIPTION:
                              0000     ;
                              0000     ; THIS ROUTINE IS ENTERED WHEN THE UNIT IS IDLE AND THERE IS A PACKET TO PROCESS.
                              0000     ;
                              0000     ; INPUTS:
                              0000     ;
                              0000     ;        I/O PACKET FORMATTED AS DESCRIBED IN TTYFDT.
                              0000     ;
                              0000     ;        R3 = I/O PACKET ADDRESS
                              0000     ;        R5 = LOGICAL UCB ADDRESS
                              0000     ;
                              0000     ; OUTPUTS:
                              0000     ;
                              0000     ;        NONE
                              0000     ;--
                              0000     TTY$STARTIO::                              ; START TERMINAL I/O
                              0000
                              0000     ;        MOVE TO PHYSICAL UCB CONTEXT. THIS INVOLVES DUPLICATING
                              0000     ;        MANIPULATIONS TO THE LOGICAL UCB DONE BY IOC$INITIATE
                              0000
          50    00A0 C5   D0  0000          MOVL    UCB$L_TL_PHYUCB(R5),R0    ; GET PHYSICAL UCB ADDRESS
             58 A0    53   D0  0005          MOVL    R3,UCB$L_IRP(R0)          ; COPY IRP ADDRESS TO PHYS UCB
          78 A0    2C A3   7D  0009          MOVQ    IRP$L_SVAPTE(R3),UCB$L_SVAPTE(R0)
       64 A0    0048 8F   AA  000E          BICW    #UCB$M_CANCEL!UCB$M_TIMOUT,UCB$W_STS(R0)
                55    50   D0  0014          MOVL    R0,R5                     ; SWITCH TO PHYSICAL UCB
                              0017
                  FFE6'   30  0017          BSBW    TTY$LOCK                  ; SETUP IPL AND REGISTERS
     54    20 A3   FFC0 8F   AB  001A          BICW3   #^C<IO$M_FCODE>,IRP$W_FUNC(R3),R4; GET INTERNAL FUNCTION CODE
                              0021          CASE    R4,TYPE=B,<DO_READ,DO_WRITE,DO_SETM,DO_SETC,DO_HANGUP,-
                              0021                          DO_MAINT,DO_HANGUP,DO_CONNECT,DO_DISCONNECT>
                              0037
```

```
                                  0037         .sbttl  START_IO ACTION ROUTINES
                                  0037
                                  0037   ;     CONNECT THIS PUCB TO A DETACHED LUCB
                                  0037
                                  0037   DO_CONNECT:
                        0C    E1  0037         BBC     #IO$V_TT_DISCON,-
                  08 20 A3        0039                 IRP$W_FUNC(R3),10$                     ;SKIP UNLESS DISCONNECT SPECIFIED
  48 A5    00020000 8F    CA  003C         BICL    #TT2$M_DISCONNECT,UCB$L_DEVDEPND2(R5); FORCE HANGUP TO COMMAND PROCE
                                  0044   10$:
        51     0084 C5    D0  0044         MOVL    UCB$L_PDT(R5),R1                       ; GET TARGET LUCB ADDRESS
                        51    13  0049         BEQL    25$                                    ; NONE, MUST BE JUST DELETED
                                  004B
                                  004B         SET_STATE        RECONNECT                     ; SET RECONNECT STATE TO TARGET LUCB
                                  004F
                        0A    BB  004F         PUSHR   #^M<R1,R3>                             ; SAVE IRP ADDRESS AND LUCB
                 54     02    D0  0051         MOVL    #TTY$V_FD_DISCONNECT,R4                ; SCHEDULE DISCONNECT COMMAND PUCB
                     FFA9'    30  0054         BSBW    TTY$CRE_FORK
                        0A    BA  0057         POPR    #^M<R1,R3>                             ; SAVE IRP ADDRESS AND LUCB
                                  0059
                                  0059   ;     NOW CONVERT IRP INTO SET MODE AND FILL IN VALUES FROM
                                  0059   ;     THE TARGET LUCB
                                  0059
        40 A3     44 A1    7D  0059         MOVQ    UCB$L_DEVDEPEND(R1),IRP$Q_TT_STATE(R3)
        39 A3     41 A1    90  005E         MOVB    UCB$B_DEVTYPE(R1),IRP$L_MEDIA+1(R3)        ; TERMINAL TYPE
        3A A3     42 A1    B0  0063         MOVW    UCB$W_DEVBUFSIZ(R1),IRP$L_MEDIA+2(R3)      ; WIDTH
                  4C A3    D4  0068         CLRL    IRP$W_TT_PRMPT(R3)
                009C C3    D4  006B         CLRL    IRP$L_VAL5(R3)
                                  006F
  40 A3    00200000 8F    CA  006F         BICL    #TT$M_MODEM,IRP$Q_TT_STATE(R3)            ; TRACK MODEM TO BE SAME AS
     08 44 A5      15    E1  0077         BBC     #TT$V_MODEM,UCB$L_DEVDEPEND(R5),20$       ;
  40 A3    00200000 8F    C8  007C         BISL    #TT$M_MODEM,IRP$Q_TT_STATE(R3)            ;
                                  0084   20$:
  44 A3    00020000 8F    CA  0084         BICL    #TT2$M_DISCONNECT,IRP$Q_TT_STATE+4(R3)    ; TRACK DISCONNECT TO BE SAM
     08 48 A5      11    E1  008C         BBC     #TT2$V_DISCONNECT,UCB$L_DEVDEPND2(R5),22$;
  44 A3    00020000 8F    C8  0091         BISL    #TT2$M_DISCONNECT,IRP$Q_TT_STATE+4(R3)  ;
                                  0099   22$:
                                  0099
                                  0099
                      0125    31  0099         BRW     DO_SET                                 ; NOW INVOKE SET MODE ACTION ROUTINE
                                  009C
        50     0908 8F    3C  009C   25$:  MOVZWL  #SS$_NOSUCHDEV,R0                      ; INDICATE DEVICE NOT AVAILABLE
                      0691    31  00A1         BRW     TTY$DONE
```

```
                    00A4   360
                    00A4   361
                    00A4   362 ;       DISCONNECT COMMAND LUCB FROM PUCB.
                    00A4   363 ;       IF NOT DETACHED, HANGUP SIGNALED TO COMMAND PROCESS
                    00A4   364
                    00A4   365 DO_DISCONNECT:
          53    DD  00A4   366         PUSHL   R3                              ; SAVE IRP
     54   02    D0  00A6   367         MOVL    #TTY$V_FD_DISCONNECT,R4         ; SCHEDULE DISCONNECT ON THAT PUCB
       FF54'    30  00A9   368         BSBW    TTY$CRE_FORK
          53  8ED0  00AC   369         POPL    R3                              ; RESTORE IRP
     50   01    3C  00AF   370         MOVZWL  #SS$_NORMAL,R0
        0680    31  00B2   371         BRW     TTY$DONE
```

```
                  00B5   373
                  00B5   374
                  00B5   375 ;          PROCESS HANGUP FUNCTION. THIS ROUTINE FORCES A MODEM HANGUP
                  00B5   376
                  00B5   377 DO_HANGUP:
         38   BB  00B5   378            PUSHR   #^M<R3,R4,R5>            ; SAVE REGISTERS
      FF46'   30  00B7   379            BSBW    CLASS_MODEM_DIS          ; DISCONNECT UNIT
         38   BA  00BA   380            POPR    #^M<R3,R4,R5>            ; RESTORE REGITERS
       0363   31  00BC   381            BRW     DO_EXIT
                  00BF   382
```

```
                      00BF
                      00BF   ;          PROCESS MAINTENANCE FUNCTIONS
                      00BF
                      00BF   DO_MAINT:
            15    E0  00BF              BBS     #TT$V_MODEM,-                           ; DISALLOW IF MODEM LINE
      41 44 A5        00C1                      UCB$L_DEVDEPEND(R5),30$
54    0118 C5     D0  00C4              MOVL    UCB$L_TT_PORT(R5),R4                    ; ACCESS PORT VECTOR
            04    E0  00C9              BBS     #IO$V_SET_MODEM,-                       ; BRANCH IF SET MODEM FUNCTION
      24 20 A3        00CB                      IRP$W_FUNC(R3),20$
      06    07    EF  00CE              EXTZV   #IO$V_LOOP,#<IO$V_LOOP_EXT-IO$V_LOOP+1>,-
50    20 A3           00D1                      IRP$W_FUNC(R3),R0                       ; GET MAINT SUBMODIFIERS
012A C5     50    88  00D4              BISB    R0,UCB$B_TT_MAINT(R5)                   ; PASS TO PORT
            50    D4  00D9              CLRL    R0                                      ; ASSUME ERROR, FOR NULL POST ROUTIN
         FF22'    30  00DB              BSBW    TTY$MAINT                               ; INVOKE PORT DRIVER TO DO FUNCTION
         7F 8F    8A  00DE              BICB    #^C<UCB$M_TT_DSBL>,-                    ; RESET ALL BUT DISABLE
      012A C5         00E1                      UCB$B_TT_MAINT(R5)
         1E 50    E9  00E4              BLBC    R0,30$                                  ; FAILURE
50    0118 C5     D0  00E7              MOVL    UCB$L_TT_PORT(R5),R0                    ; GET PORT VECTOR ADDRESS
         FF11'    30  00EC              BSBW    TTY$RESUME                              ; RESET ANY CONTROLS STATE
         0330     31  00EF              BRW     DO_EXIT                                 ; SUCCESS
                      00F2   20$:                                                       ; PROCESS SET MODEM SIGNALS
   52    3A A3    3C  00F2              MOVZWL  IRP$L_MEDIA+2(R3),R2                    ; GET SET/RESET MODEM MASK
   52    E5 8F    8A  00F6              BICB    #^C<TT$M_DS_DTR!-                       ; CLEAR ALL BUT MODEM OUTPUT
                      00FA                      TT$M_DS_SECTX!-                         ; BITS
                      00FA                      TT$M_DS_RTS>,R2
            53    DD  00FA              PUSHL   R3                                      ; SAVE VOLITAL REGISTER
         FF01'    30  00FC              BSBW    TTY$DS_SET                              ; SET /RESET SPECIFIED SIGNALS
         53 8ED0      00FF              POPL    R3                                      ; RESTORE REGISTER
         031D     31  0102              BRW     DO_EXIT                                 ; SUCCESS EXIT
                      0105   30$:
      50    2C    3C  0105              MOVZWL  #SS$_ABORT,R0                           ; ERROR EXIT
            51    D4  0108              CLRL    R1
         0628     31  010A              BRW     TTY$DONE
```

```
                          010D    ;
                          010D    ; READ OPERATION
                          010D    ;
                          010D    DO_READ:                                                ;
     54    78 A5    D0    010D                    MOVL    UCB$L_SVAPTE(R5),R4           ; GET THE ADDRESS OF THE READ PACKET
                          0111
     62    40 A3    C8    0111                    BISL    IRP$Q_TT_STATE(R3),(R2)       ; Set the read state bits.
  04 A2    44 A3    C8    0115                    BISL    IRP$Q_TT_STATE+4(R3),4(R2)
                          011A
           38 A3    D4    011A                    CLRL    IRP$L_MEDIA(R3)               ; Set up storage for the read
                          011D                                                          ; terminator.
           04       AA    011D                    BICW    #UCB$M_TT_NOTIF,-             ; Set the "user has not been
           68 A5          011F                            UCB$W_DEVSTS(R5)              ; notified" bit.
           0B       E1    0121                    BBC     #IO$V_PURGE,-                 ; Branch forward if purge type-
     03 20 A3             0123                            IRP$W_FUNC(R3),10$            ; ahead not requested.
        FED7'       30    0126                    BSBW    TTY$PURGE_AHEAD               ; Otherwise, purge buffer.
                          0129                                                          ; the write completes.
                          0129    10$:
      00E4 C5       D5    0129                    TSTL    UCB$L_TT_TYPAHD(R5)           ; Type ahead buffer allocated?
           73       13    012D                    BEQL    30$                           ; Not yet
                          012F
        FECE'       30    012F    12$:            BSBW    TTY$SETUP_READ                ; Set up the UCB for a read
                          0132                                                          ; operation.
                          0132
                          0132    ;
                          0132    ; CHECK FOR LINE FEED NEEDED
                          0132    ;
                          0132    ;
                          0132                    IF_STATE -                            ; Skip if passall, or
                          0132                    <PASALL>,25$
                          0136                    IF_NOT_STATE NOECHO,14$               ; NO ECHO THEN
                          013A                    CLR_STATE EDITING                     ; NO EDITING
                          013E    14$:            IF_STATE -
                          013E                    <NL,WRAP>,20$                         ; if already did line feed.
                          0145
                          0145                    IF_NOT_STATE NOECHO,15$               ; Branch if echo
           00       E1    0149                    BBC     #TT2$V_LOCALECHO,-            ; Branch if not local echo
        26 48 A5          014B                            UCB$L_DEVDEPND2(R5),20$
                          014E
      00FC C5       B5    014E    15$:            TSTW    UCB$W_TT_CURSOR(R5)           ; CURSOR AT 0?
           20       12    0152                    BNEQ    20$                           ; If no, send no line feed.
      00FF C5       91    0154                    CMPB    UCB$B_TT_LASTC(R5),-          ; Was the last character also a
           0D             0158                            #TTY$C_CR                     ; carriage return?
           19       12    0159                    BNEQ    20$                           ; No. Don't send free linefeed.
                          015B                    IF_NOT_STATE SKIPLF,17$               ; NO SKIP LINEFEED THEN BYPASS NOCRLF CHECK
 0F 0122 C5    07   E0    015F                    BBS     #TTY$V_PC_NOCRLF,UCB$W_TT_PRTCTL(R5),20$;SO JUST ECHO THE CHARACTER
                          0165    17$:            SET_STATE <SENDLF>                    ; SEND A LINE FEED FIRST
                          0168                    CLR_STATE <SKIPLF>
                          016C                    IF_NOT_STATE PROMPT,20$               ; DO WE HAVE A
                          0170                    SET_STATE <SKIPLF>                    ; SEND A LINE FEED FIRST
                          0174    20$:            IF_STATE RDVERIFY,25$                 ; THIS ISN'T NECESSARY IF READ VERIFY
                          0178
     54    78 A5    D0    0178                    MOVL    UCB$L_SVAPTE(R5),R4           ; GET THE READ PACKET ADDRESS
     7E  00FC C5    3C    017C                    MOVZWL  UCB$W_TT_CURSOR(R5),-(SP); SAVE THE CURSOR POSITION FOR ECHOING
     02    44 A4    B1    0181                    CMPW    TTY$W_RB_MODE(R4),#TTY$K_ER_ECHLINE; IS THIS A READ WITH INITIAL
                          0185                                                          ; OFFSET.
           0D       12    0185                    BNEQ    21$                           ; NO THEN USE NORMAL
```

```
00FC C5   012B C5   9B  0187        MOVZBW  UCB$B_TT_OLDCPZORG(R5),UCB$W_TT_CURSOR(R5); YES THEN USE THE
                        018E                                        ; STORED INITIAL CURSOR POSITION
  3A A4   012B C5   9B  018E        MOVZBW  UCB$B_TT_OLDCPZORG(R5),TTY$W_RB_CPZORG(R4);
              32 A4 B4  0194   21$: CLRW    TTY$W_RB_LINREST(R4)    ; NO EXTRA CHARACTERS
              FE66'  30 0197        BSBW    FIND_BOL_NOCLEAR        ; find the wrapping and
     00FC C5     8E  F7 019A        CVTLW   (SP)+,UCB$W_TT_CURSOR(R5); RESTORE THE CURSOR POSITION FOR ECHOING
               0390  31 019F   25$: BRW     TTY$STARTOUTPUT         ; Go start the read.
                        01A2   30$:
                        01A2        ; Allocate typeahead buffer
                        01A2
                  53 DD 01A2        PUSHL   R3                      ; SAVE IRP ADDRESS
                  53 D4 01A4        CLRL    R3                      ; INDICATE NO DATA
              54  01 D0 01A6        MOVL    #TTY$V_FD_GETAHD,R4     ; ASK FOR TYPEAHD FORK
              FE54'  30 01A9        BSBW    TTY$CRE_FORK            ; GO ALLOCATE BUFFER
                53 8ED0 01AC        POPL    R3                      ; RESTORE IRP ADDRESS
               FF7D  31 01AF        BRW     12$                     ; CONTINUE PROCESSING
                        01B2
```

```
                          01B2   493
                          01B2   494 ; SET MODE OPERATION -
                          01B2   495
                          01B2   496 DO_SETM:
            000C   31     01B2   497          BRW     DO_SET
                          01B5   498
                          01B5   499 ; DO SET CHARACTERISTICS
                          01B5   500
                          01B5   501 DO_SETC:                                 ; DO PRIVILEGED SET
        32 A3   0C   B1   01B5   502          CMPW    #12,IRP$W_BCNT(R3)       ; CHECK PARAMETERS
                06   15   01B9   503          BLEQ    DO_SET                   ; ALL SPECIFIED
           00C8 C5   D0   01BB   504          MOVL    UCB$L_TT_DECHA1(R5),-    ; INIT DEFAULT IF NOT SPECIFIED
             44 A3        01BF   505                  IRP$Q_TT_STATE+4(R3)
                          01C1   506
                          01C1   507 ; PROCESS CHANGE OF CHARACTERISTICS AND MODE
                          01C1   508 ; CHANGE BASIC MODE BITS IN UCB$L_DEVDEPEND
                          01C1   509
                          01C1   510 DO_SET:
 54    44 A5   40 A3   CD 01C1   511          XORL3   IRP$Q_TT_STATE(R3),UCB$L_DEVDEPEND(R5),R4; GET MODIFIED BITS
          08 54   0D   E5 01C7   512          BBCC    #TT$V_REMOTE,R4,8$       ; DISALLOW CLEARING REMOTE BIT
 40 A3   00002000 8F   CA 01CB   513          BICL    #TT$M_REMOTE,IRP$Q_TT_STATE(R3)
                          01D3   514                                           ; DISALLOW SETTING REMOTE BIT
                          01D3   515 8$:
        44 A5   54   CA   01D3   516          BICL    R4,UCB$L_DEVDEPEND(R5)   ; CLEAR THE CHANGED BITS
     44 A5   40 A3   C8   01D7   517          BISL    IRP$Q_TT_STATE(R3),UCB$L_DEVDEPEND(R5);
     41 A5   39 A3   90   01DC   518          MOVB    IRP$L_MEDIA+1(R3),UCB$B_DEVTYPE(R5); INSERT NEW TERMINAL TYPE
                          01E1   519 ;
                          01E1   520 ;        DEVDP1 BIT CHECKING
                          01E1   521 ;
        50   44 A3   D0   01E1   522          MOVL    IRP$Q_TT_STATE+4(R3),R0  ; GET SECOND DEVDEPEND WORD
   51   48 A5   50   CD   01E5   523          XORL3   R0,UCB$L_DEVDEPND2(R5),R1 ; GET MODIFIED BITS
 50    00000200 8F   CA   01EA   524          BICL    #<TT2$M_DCL_MAILBX>,R0   ; REMOVE DCL SPECIFIC BITS
                          01F1   525
        24 51   06   E1   01F1   526          BBC     #TT2$V_DMA,R1,12$        ; SKIP IF DMA NOT CHANGED
        14 50   06   E1   01F5   527          BBC     #TT2$V_DMA,R0,10$        ; BRANCH IF TURNING DMA OFF
   0E 0122 C5   02   E1   01F9   528          BBC     #TTY$V_PC_DMAAVL,UCB$W_TT_PRTCTL(R5),10$ ; DONT IF FEATURE NOT AVAIL
 50    00000040 8F   C8   01FF   529          BISL    #TT2$M_DMA,R0            ; SET DMA CHARACTERISTIC ON
      0122 C5   02   A8   0206   530          BISW    #TTY$M_PC_DMAENA,UCB$W_TT_PRTCTL(R5) ; ENABLE IN PORT
                0C   11   020B   531          BRB     12$
      0122 C5   02   AA   020D   532 10$:     BICW    #TTY$M_PC_DMAENA,UCB$W_TT_PRTCTL(R5) ; DISABLE DMA IN PORT
 50    00000040 8F   CA   0212   533          BICL    #TT2$M_DMA,R0            ; RESET DMA CHARACTERISTIC
                          0219   534
                          0219   535 12$:     NOMOD   ALTYPEAHD                ; DISALLOW CHANGING TYPE AHEAD
                          0224   536
                03   E0   0224   537          BBS     #TT2$V_MODHANGUP,-
          11 48 A5        0226   538                  UCB$L_DEVDEPND2(R5),15$  ; BRANCH IF MODIFY HANGUP ALLOWED
                          0229   539
                          0229   540          PRIV_TO_MOD HANGUP               ; REQUIRE PRIV TO MODIFY HANGUP
                          023A   541 15$:     PRIV_TO_MOD SETSPEED             ; REQUIRE PRIV TO MODIFY SET SPEED
                          024B   542          PRIV_TO_MOD SECURE               ; REQUIRE PRIVS TO MODIFY SECURE SERVER
                          025C   543          PRIV_TO_MOD MODHANGUP            ; REQUIRE PRIV TO MODIFY MODHANGUP
                          026D   544
        06 50   05   E1   026D   545          BBC     #TT2$V_XON,R0,20$        ; BRANCH IF NO XON REQUESTED
           50   20   CA   0271   546          BICL    #TT2$M_XON,R0            ; RESET XON BIT.
             FD89'   30   0274   547          BSBW    TTY$RESUME               ; CALL RESUME
                          0277   548
                          0277   549 20$:
```

```
            48 A5     50   D0 0277           MOVL    R0,UCB$L_DEVDEPND2(R5) ; SET SECOND DEVDEPENDENT WORD
                              027B      ;
                              027B      ; SET UP WIDTH
                              027B      ;
         42 A5     3A A3   B0 027B           MOVW    IRP$L_MEDIA+2(R3),UCB$W_DEVBUFSIZ(R5); INSERT NEW CARRIAGE WIDTH
                              0280      ;
                              0280      ; SET UP SPEED
                              0280      ;
            51     4C A3   3C 0280           MOVZWL  IRP$W_TT_PRMPT(R3),R1     ; GET NEW SPEED
                      1E   13 0284           BEQL    30$                       ; IF EQL THEN NO CHANGE
                              0286      ;
                              0286      ; SET SPEED PRIVILEGE CHECK
                              0286      ;
          00F4 C5     51   91 0286           CMPB    R1,UCB$W_TT_SPEED(R5)     ; IS LOW ORDER BYTE OF SPEED CHANGING?
                      12   13 028B           BEQL    28$                       ; BRANCH IF NOT
                              028D
                      08   E1 028D           BBC     #TT2$V_SETSPEED,-
                   0D 48 A5   028F                   UCB$L_DEVDEPND2(R5),28$ ; BRANCH IF SET SPEED ALLOWED
      58 B3    00400080 8F D3 0292           BITL    #<<1@PRV$V_LOG_IO>! -     ; DOES PROCESS HAVE LOG_IO
                              029A                   <1@PRV$V_PHY_IO>>, -      ; OR PHY_IO PRIVILEGE?
                              029A                   @IRP$L_ARB(R3)            ; CHECK ACCESS RIGHTS BLOCK
                      03   12 029A           BNEQ    28$                       ; BRANCH IF PRIVILEGED
                    019D   31 029C           BRW     NOPRIV_EXIT               ; PRIV FAILURE
                              029F
                              029F      ;
                              029F      ; PROCESS PARITY SETTINGS
                              029F      ;
          00F4 C5     51   B0 029F 28$:      MOVW    R1,UCB$W_TT_SPEED(R5)     ; INSERT LINE SPEED
       12 009C C3     05   E1 02A4 30$:      BBC     #TT$V_ALTRPAR,IRP$L_VAL5(R3),35$; BR IF PARITY SHOULD NOT BE ALTERED
    50    009C C3   3F 8F  8B 02AA           BICB3   #^C<TT$M_PARITY!TT$M_ODD>,IRP$L_VAL5(R3),R0; RESET BITS
          00F8 C5   C0 8F  8A 02B1           BICB    #TT$M_PARITY!TT$M_ODD,UCB$B_TT_PARITY(R5); CLEAR CURRENT PARITY
          00F8 C5     50   88 02B7           BISB    R0,UCB$B_TT_PARITY(R5)    ; INSERT NEW VALUE
                              02BC      ;
                              02BC      ; SET UP CHARACTER SIZE AND STOP BITS
                              02BC      ;
                              02BC 35$:
       08 009C C3     04   E0 02BC           BBS     #TT$V_ALTFRAME,IRP$L_VAL5(R3),36$; DOES THE USER WANT A NEW FRPAM SI
       21 00F8 C5     02   E1 02C2           BBC     #UCB$V_TT_USERFRAME,UCB$B_TT_PARITY(R5),37$; DID THE USER SPECIFY
                              02C8                                             ; THE FRAME SIZE?
                      3B   11 02C8           BRB     42$                       ; YES THEN DON'T BOTHER IT
       00 00F8 C5     02   E5 02CA 36$:      BBCC    #UCB$V_TT_USERFRAME,UCB$B_TT_PARITY(R5),38$
50     009C C3   FFFFFFF0 8F CB 02D0 38$:    BICL3   #^C<^X0F>,IRP$L_VAL5(R3),R0; GET THE NEW FRAME SIZE
                      0D   13 02DA           BEQL    37$                       ; 0 SPECIFYED THEN CLEAR USER FRAME
   00F8 C5   02    03   50 F0 02DC           INSV    R0,#UCB$V_TT_LEN,#2,UCB$B_TT_PARITY(R5); SET THE
                              02E3                                             ; PARITY CORRECTLY
       1C 00F8 C5     02   E3 02E3           BBCS    #UCB$V_TT_USERFRAME,UCB$B_TT_PARITY(R5),42$; AND SETUSER FRAME
                              02E9                                             ; SPECIFYED THEN CONTINUE ON
                      06   E1 02E9 37$:      BBC     #TT$V_PARITY,-
             11 00F8 C5       02EB                   UCB$B_TT_PARITY(R5),40$ ; IF NO PARITY, USE 8 BIT
                      0F   E0 02EF           BBS     #TT$V_EIGHTBIT,-
                 0C 44 A5     02F1                   UCB$L_DEVDEPEND(R5),40$ ; USE 8 BIT SIZE
          00F8 C5     18   8A 02F4           BICB    #UCB$M_TT_LEN,UCB$B_TT_PARITY(R5)        ; RESET CHARACTER FRAME
          00F8 C5     10   88 02F9           BISB    #^X10,UCB$B_TT_PARITY(R5)       ; SET 7 BIT CHARACTER FRAME
                      05   11 02FE           BRB     42$
                              0300 40$:
          00F8 C5     18   88 0300           BISB    #UCB$M_TT_LEN,UCB$B_TT_PARITY(R5)        ; SET 8 BIT CHARACTER FRAME
                              0305 42$:
```

```
  11 009C C3    0A    E1 0305                 BBC     #TT$V_ALTDISPAR,IRP$L_VAL5(R3),41$      ; CHECK FOR DISABLE PARITY E
     00F8 C5    02    CA 030B                 BICL    #UCB$M_TT_DISPARERR,UCB$B_TT_PARITY(R5) ; CLEAR DISMISS
  06 009C C3    09    E1 0310                 BBC     #TT$V_DISPARERR,IRP$L_VAL5(R3),41$      ; DOES HE WANT IT SET
  00 00F8 C5    01    E2 0316                 BBSS    #UCB$V_TT_DISPARERR,UCB$B_TT_PARITY(R5),41$; YES THEN SET IT
  07 009C C3    08    E0 031C         41$:    BBS     #TT$V_TWOSTOP,IRP$L_VAL5(R3),43$        ; DOES HE WANT TWO STOP BITS
  04    00F4 C5    91 0322                    CMPB    UCB$W_TT_SPEED(R5),#4                   ; SPEED <= 150 BAUD?
              07    14 0327                   BGTR    44$                                     ; NO
     00F8 C5    20    88 0329         43$:    BISB    #UCB$M_TT_STOP,UCB$B_TT_PARITY(R5)      ; FLAG 2 STOP BITS
                05    11 032E                 BRB     46$
                         0330         44$:
     00F8 C5    20    8A 0330                 BICB    #UCB$M_TT_STOP,UCB$B_TT_PARITY(R5)      ; FLAG 1 STOP BIT
                         0335         ;
                         0335         ; PROCESS FILL DATA
                         0335         ;
                         0335         46$:
                50    D4 0335                 CLRL    R0                              ; ASSUME NEW VALUE IS 0
  0A 44 A5      0A    E1 0337                 BBC     #TT$V_CRFILL,UCB$L_DEVDEPEND(R5),50$; CR FILL ON?
  50    4E A3         9A 033C                 MOVZBL  IRP$W_TT_PRMPT+2(R3),R0 ; GET NEW VALUE
  02 54         0A    E0 0340                 BBS     #TT$V_CRFILL,R4,50$     ; CHANGE?
                0C    13 0344                 BEQL    60$                     ; IF NEQ EQL 0 THEN NO CHANGE
     00F6 C5    50    90 0346         50$:    MOVB    R0,UCB$B_TT_CRFILL(R5)  ; RESET VALUE
                05    12 034B                 BNEQ    60$                     ; IF NEQ THEN OK
  00 44 A5      0A    E5 034D                 BBCC    #TT$V_CRFILL,UCB$L_DEVDEPEND(R5),60$; SET OFF
                50    D4 0352         60$:    CLRL    R0                      ; ASSUME NEW VALUE IS 0
  0A 44 A5      0B    E1 0354                 BBC     #TT$V_LFFILL,UCB$L_DEVDEPEND(R5),65$; LF FILL ON?
  50    4F A3         9A 0359                 MOVZBL  IRP$W_TT_PRMPT+3(R3),R0 ; GET NEW VALUE
  02 54         0B    E0 035D                 BBS     #TT$V_LFFILL,R4,65$     ; CHANGE?
                0C    13 0361                 BEQL    75$                     ; IF NEQ EQL 0 THEN NO CHANGE
     00F7 C5    50    90 0363         65$:    MOVB    R0,UCB$B_TT_LFFILL(R5)  ; RESET VALUE
                05    12 0368                 BNEQ    75$                     ; IF NEQ THEN OK
  00 44 A5      0B    E5 036A                 BBCC    #TT$V_LFFILL,UCB$L_DEVDEPEND(R5),75$; SET OFF
                         036F         ;
                         036F         ; CHECK FOR CHANGE IN STATUS OF MODEM
                         036F         ;
                         036F         75$:
     12 54      15    E1 036F                 BBC     #TT$V_MODEM,R4,80$              ; NO CHANGE IN MODEM STATUS
                1F    BB 0373                 PUSHR   #^M<R0,R1,R2,R3,R4>             ; SAVE VOLITAL REGISTERS
        51      00    9A 0375                 MOVZBL  #MODEM$C_INIT,R1                ; ASSUME MODEM INIT
     44 A5      15    E0 0378                 BBS     #TT$V_MODEM,UCB$L_DEVDEPEND(R5),-
                03       037C                         77$                             ; IT IS INIT
        51      01    9A 037D                 MOVZBL  #MODEM$C_SHUTDWN,R1             ; NO, SHUT DOWN
                         0380         77$:
           FC7D'  30 0380                     BSBW    TRANSITION_NOCHECK              ; DECLARE MODEM TRANSITION
                1F    BA 0383                 POPR    #^M<R0,R1,R2,R3,R4>
                         0385
                         0385         80$:
                         0385         ;
                         0385         ; enable or disable AUTO XON AND XOFF
                         0385         ;
  0122 C5     0040 8F     A8 0385             BISW    #TTY$M_PC_XOFENA,UCB$W_TT_PRTCTL(R5); TURN ON AUTO XOFF
    05 44 A5      00     E0 038C             BBS     #TT$V_PASSALL,UCB$L_DEVDEPEND(R5),84$; IS THIS PASSALL? YES THEN
    07 44 A5      05     E0 0391             BBS     #TT$V_TTSYNC,UCB$L_DEVDEPEND(R5),85$; IF TT SYNC IS SET THEN LEAVE I
  0122 C5     0040 8F     AA 0396     84$:    BICW    #TTY$M_PC_XOFENA,UCB$W_TT_PRTCTL(R5); TURN OFF AUTO XOFF
                         039D         85$:
                         039D         ;
                         039D         ; COPY OVER PASSALL AND NOECHO TO CURRENT STATE
                         039D         ;
```

```
          54  44 A5   FC 8F  8B 039D          BICB3   #^C<TT$M_PASSALL!TT$M_NOECHO>,UCB$L_DEVDEPEND(R5),R4
        04 A2   02    02  54  F0 03A3          INSV    R4,#TTY$V_ST_PASALL,#2,4(R2); INSERT IN STATE VECTOR
        54   44 A5    01  03  EE 03A9          EXTV    #TT$V_ESCAPE,#1,UCB$L_DEVDEPEND(R5),R4  ; GET CURRENT SETTING
        04 A2   01    0B  54  F0 03AF          INSV    R4,#TTY$V_ST_ESCAPE,#1,4(R2)            ; UPDATE IN STATE
             04 48 A5     12  E1 03B5          BBC     #TT2$V_PASTHRU,UCB$L_DEVDEPND2(R5),98$  ; IN PASS THRU MODE
                                 03BA          SET_STATE PASALL
                                 03BE  98$:
                                 03BE  ;
                                 03BE  ; INIT THE UNIT TO CHANGE THE SPEED AND PARITY
                                 03BE  ;
                     FC3F'   30  03BE          BSBW    TTY$SET_LINE                    ; INIT LINE SPEED AND PARITY
                                 03C1  ;
                                 03C1  ; IF MULTI IS SET THEN THE I/O MUST BE STARTED
                                 03C1  ;
                                 03C1  100$:   IF_NOT_STATE MULTI,110$         ; BR IF MULTI NO SET
                     016A    30  03C5          BSBW    TTY$STARTOUTPUT                 ; START THE MULTIPLE OUTPUT
                                 03C8  ;
                                 03C8  ; CHECK FOR SET CHARACTERISTICS AND RETURN IOSB DATA
                                 03C8  ;
               53    58 A5   D0  03C8  110$:   MOVL    UCB$L_IRP(R5),R3                ; GET CURRENT PACKET ADDRESS
        03    20 A3    06    00  ED  03CC      CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3),#TTY$C_FC_SETC; SET CHAR?
                          3A  12 03D2          BNEQ    120$                            ; IF NEQ THEN NO
          00E8 C5    00F4 C5  D0 03D4          MOVL    UCB$W_TT_SPEED(R5),UCB$W_TT_DESPEE(R5); RESET PERM SPEED
                          01  E1 03DB          BBC     #TT2$V_AUTOBAUD,-
                  05 48 A5       03DD                  UCB$L_DEVDEPND2(R5),115$ ; BRANCH IF NOAUTOBAUD
                          0F  9B 03E0          MOVZBW  #TT$C_BAUD_9600,-
                     00E8 C5     03E2                  UCB$W_TT_DESPEE(R5)     ; SET PERMANENT 9600 BAUD FOR AUTOBAUD
          00EC C5    00F8 C5  90 03E5  115$:   MOVB    UCB$B_TT_PARITY(R5),UCB$B_TT_DEPARI(R5); RESET PERM PARITY
            00F0 C5    41 A5  D0 03EC          MOVL    UCB$B_DEVTYPE(R5),UCB$B_TT_DETYPE(R5); RESET TYPE AND WIDTH
  00C4 C5    44 A5  00002000 8F  CB 03F2       BICL3   #TT$M_REMOTE,UCB$L_DEVDEPEND(R5),UCB$L_TT_DECHAR(R5); RESET PERM CHA
            00C8 C5    48 A5  D0 03FD          MOVL    UCB$L_DEVDEPND2(R5),UCB$L_TT_DECHA1(R5) ;UPDATE SECOND CHAR WORD
                                 0403
             06 44 A3     07  E1 0403          BBC     #TT2$V_ALTYPEAHD,IRP$Q_TT_STATE+4(R3),120$; SKIP IF ALTERNATE
                                 0408                                                  ; TYPEAHEAD NOT SPECIFIED
          00 00C8 C5      07  E2 0408          BBSS    #TT2$V_ALTYPEAHD,UCB$L_TT_DECHA1(R5),120$ ; ONLY ALLOW SETTING
                                 040E                                                  ; AS PERM CHARACTERISTIC
                                 040E  120$:
                                 040E
                                 040E  ;       UPDATE NEWLY WRITTEN FIELDS WHICH ARE MAINTAINED
                                 040E  ;       IN BOTH THE LOGICAL ANY PHYSICAL UCB
                                 040E
               50    00C0 C5  D0 040E          MOVL    UCB$L_TT_LOGUCB(R5),R0                  ; GET LOGICAL UCB ADDRESS
             44 A0     44 A5  D0 0413          MOVL    UCB$L_DEVDEPEND(R5),UCB$L_DEVDEPEND(R0) ; UPDATE CHARACTERISTICS
             48 A0     48 A5  D0 0418          MOVL    UCB$L_DEVDEPND2(R5),UCB$L_DEVDEPND2(R0) ; "UCB$L_TT_DEVDP1"
             40 A0     40 A5  D0 041D          MOVL    UCB$B_DEVCLASS(R5),UCB$B_DEVCLASS(R0)   ; CLASS,TYPE,BUFSIZE
                                 0422
                                 0422  ;       THIS ROUTINE COMPLETES SET AND SENSE CHARACTERISTICS OPERATIONS
                                 0422  ;       AND RETURNS STATUS VALUES IN THE IOSB
                                 0422
                                 0422  DO_EXIT:
               50    00F2 C5  D0 0422          MOVL    UCB$W_TT_SPEED-2(R5),R0; RETURN SPEED
    51    00F6 C5  FF3FFFFF 8F  CB 0427        BICL3   #^C<<UCB$M_TT_PARTY!UCB$M_TT_ODD>@16>,UCB$B_TT_PARITY-2(R5),R1;
               51    00F6 C5  B0 0431          MOVW    UCB$B_TT_CRFILL(R5),R1  ; INSERT FILL DATA
                  50      01  B0 0436          MOVW    #SS$_NORMAL,R0          ; SET STATUS
                        02F9  31 0439          BRW     TTY$DONE
                                 043C
                                 043C  NOPRIV_EXIT:
```

```
           50   24   D0 043C   721        MOVL    #SS$_NOPRIV,R0        ; SET NO PRIV
                51   D4 043F   722        CLRL    R1                    ; CLEAR R1
              02F1   31 0441   723        BRW     TTY$DONE
                        0444   724
```

```
                          0444   ;
                          0444   ; WRITE OPERATION
                          0444   ;
                          0444   DO_WRITE:                                                ;
                          0444   ;
                          0444   ; Control only comes here in the case of half duplex writes.  Full
                          0444   ; duplex writes use the TTY$WRTSTARTIO entry point.
                          0444   ;
         53    2C A3   D0 0444           MOVL    IRP$L_SVAPTE(R3), R3       ; Get addr of write block
          00000532'EF  9F 0448           PUSHAB  TTY$STARTOUTPUT            ; if write is started, control
                          044E                                              ; will return to STARTOUTPUT
               0040    30 044E           BSBW    WRTSTARTIO                 ; if queued, control will return
                       05 0451           RSB                                ; here.  Return to caller.
```

```
0452          .SBTTL  TTY$WRTSTARTIO - Starts or queues a write operation
0452
0452  ;++
0452  ;
0452  ; Functional description:
0452  ;
0452  ;       If called from an FDT routine (or from EXE$BRDCST),
0452  ;       TTY$WRTSTARTIO first raises to device IPL, and then calls the
0452  ;       internal routine. All other code enters through the WRTSTARTIO
0452  ;       entry point.
0452  ;
0452  ;       If a write is occurring, the routine queues the write buffer.
0452  ;       If a read is occurring, but the buffer header specifies
0452  ;               write-breakthrough, the routine starts the write.
0452  ;       If a read is occurring, but no characters have been received
0452  ;               yet, the routine starts the write.
0452  ;       Otherwise, the routine queues the write buffer.
0452  ;
0452  ;       To start the write operation, the routine writes the address
0452  ;       of the buffer in UCB$L_TT_WRTBUF, sets and clears various
0452  ;       state bits, and returns.
0452  ;
0452  ;       To queue the buffer, the routine inserts the buffer address at
0452  ;       the end of the queue unless the header specfies write-
0452  ;       breakthrough. In the latter case, the buffer address is inserted
0452  ;       after the last write-breakthrough request in the queue.
0452  ;
0452  ;       Returning from WRTSTARTIO is odd. The routine assumes that 0(SP)
0452  ;       is the address to return to if the write is to start now.
0452  ;       If the routine instead queues the write, the routine pops this
0452  ;       start-write address of the stack, and returns to the real
0452  ;       caller.
0452  ;
0452  ; Inputs:
0452  ;
0452  ;       R3      - address of the write buffer
0452  ;       R5      - address of the UCB
0452  ;
0452  ; Implicit inputs:
0452  ;
0452  ;       The write buffer consists of a header, and an optional message
0452  ;       buffer. For broadcast messages, the message buffer is absent.
0452  ;       TTY$L_WB_FR3 is 0 for a normal broadcast and 1 for and ANSI
0452  ;       broadcast or one that specified norefresh.
0452  ;
0452  ; Outputs:
0452  ;
0452  ;       If the broadcast message is rejected, the TTY$L_WB_END field
0452  ;       of the write packet is zeroed.
0452  ;
0452  ;       R0      - preserved
0452  ;       R1      - scratch
0452  ;       R2      - address of UCB state bits
0452  ;       R3      - address of buffer
0452  ;       R4      - if packet is started, address of IRP or 0
0452  ;               - if packet is queued, scratch
0452  ;       R5      - address of UCB
```

```
                                0452      ;
                                0452      ; Implicit outputs:
                                0452      ;
                                0452      ;       Buffer may be entered in queue.
                                0452      ;
                                0452      ;       If write operation is started,
                                0452      ;       UCB$L_TT_WRTBUF - address of buffer
                                0452      ;       UCB$Q_TT_STATE  - write bit, and other bits from IRP are set
                                0452      ;                         control-O may be canceled
                                0452      ;       UCB$L_DEVDEPEND - mailbox may be enabled
                                0452      ;
                                0452      ;--
                                0452
                                0452      TTY$WRTSTARTIO::                              ; Start or queue write.
          51   00A0 C5     D0   0452              MOVL    UCB$L_TL_PHYUCB(R5),R1   ; GET PUCB ADDRESS
                  22       13   0457              BEQL    30$                      ; NONE CURRENTLY EXISTS
               55 51       D0   0459              MOVL    R1,R5                    ; SWITCH TO PUCB CONTEXT
               24 A3       D5   045C              TSTL    TTY$L_WB_IRP(R3)         ; Is this a broadcast?
                  0C       13   045F              BEQL    20$                      ; YES, SPECIAL CASE
                                0461      10$:                                  ; Acceptable packet.
               FB9C'       30   0461              BSBW    TTY$LOCK                 ; Raise to DIPL, get states.
           00000532'EF     9F   0464              PUSHAB  TTY$STARTOUTPUT          ; Set up return address to
                                046A                                               ; start the output.
                  25       10   046A              BSBB    WRTSTARTIO               ; Start or queue the packet.
                           05   046C              RSB                              ; If packet queued, control
                                046D                                               ; returns here, so return to
                                046D                                               ; caller.
                                046D      ;       INTERNAL BROADCAST PACKET.
                                046D      ;       CHECK FOR DISABLE
                                046D      20$:
                           D3   046D              BITL    #TT$M_PASSALL!-          ; Test for passall and/or
                                046E                      TT$M_NOBRDCST,-          ; nobroadcast modes set in the
     44 A5    00020001 8F       046E                      UCB$L_DEVDEPEND(R5)      ; term's UCB (ignore NOECHO).
                  EA       13   0475              BEQL    10$                      ; Continue if not set.
               20 A3       D4   0477      25$:    CLRL    TTY$L_WB_END(R3)         ; Zero end address to indicate
                                047A                                               ; failure to EXE$BRDCST.
                           05   047A              RSB                              ; And return to EXE$BRDCST.
                                047B
                                047B      ;       LUCB CURRENTLY DETACHED
                                047B      ;       COMPLETE THE WRITE
                                047B              ASSUME IRP$L_IOST1+4 EQ IRP$L_IOST2
                                047B      30$:
          51   24 A3       D0   047B              MOVL    TTY$L_WB_IRP(R3),R1      ; GET IRP ADDRESS
                  F6       13   047F              BEQL    25$                      ; INTERNAL BROADCAST, REJECT IT.
               53 51       D0   0481              MOVL    R1,R3                    ; GET IRP ADDRESS
               38 A3       7C   0484              CLRQ    IRP$L_IOST1(R3)          ; INIT IOSB RETURN
            38 A3 01       B0   0487              MOVW    #SS$_NORMAL,IRP$L_IOST1(R3)
           00000000'GF     17   048B              JMP     G^COM$POST
                                0491
                                0491
                                0491
                                0491      WRTSTARTIO:                                   ; Checks for start or queue.
                                0491
                                0491      ;
                                0491      ; Inputs:
                                0491      ;
                                0491      ;       R2      - address of state bits longword
```

```
                     0491  ;            R3        - address of write packet
                     0491  ;            R5        - address of the device's UCB
                     0491  ;
                     0491  ;            0(SP)     - address to return if write is queued
                     0491  ;            4(SP)     - address to return to if write is started
                     0491  ;
                     0491
           01    BB  0491               PUSHR   #^M<R0>                   ; Save a register.
 50    00CC C5   9E  0493               MOVAB   UCB$L_TT_WFLINK(R5),R0    ; Get address of write queue.
   54    24 A3   D0  0498               MOVL    TTY$L_WB_IRP(R3),R4       ; Get address of IRP.
                     049C               IF_STATE -                        ; If writing is in progress,
                     049C                       WRITE,QUEUE_PKT           ; just queue the packet.
           54    D5  04A0               TSTL    R4                        ; See if the packet has an IRP.
           42    13  04A2               BEQL    START_PKT                 ; If not, just start the packet.
                     04A4               IF_NOT_STATE -                    ; If not in a read state,
                     04A4                       READ,START_PKT            ; go ahead and start the packet.
                     04A8               IF_STATE -                        ; If noecho read, no blocking
                     04A8                       NOECHO,START_PKT          ; so start the write
           09    E0  04AC               BBS     #IO$V_BREAKTHRU,-
     35 20 A4        04AE                       IRP$W_FUNC(R4),START_PKT; Start if break thru write.
   51    78 A5   D0  04B1               MOVL    UCB$L_SVAPTE(R5),R1       ; get the read packet address
         3C A1   B5  04B5               TSTW    TTY$W_RB_TXTOFF(R1)       ; and check if we have started typing
           2C    13  04B8               BEQL    START_PKT                 ; go start the packet.
           19    11  04BA               BRB     QUEUE_LAST                ; Otherwise, queue packet at end
                     04BC                                                 ; of queue.
                     04BC
                     04BC       QUEUE_PKT:
           54    D5  04BC               TSTL    R4                        ; If an IRP is associated,
           15    12  04BE               BNEQ    QUEUE_LAST                ; queue packet at end of queue.
   54      50    D0  04C0               MOVL    R0,R4                     ; Make a copy of queue head.
                     04C3
                     04C3       10$:
   51      64    D0  04C3               MOVL    TTY$L_WB_FLINK(R4),R1     ; Get first queue entry.
   50      51    D1  04C6               CMPL    R1,R0                     ; See if at end of queue.
           0A    13  04C9               BEQL    QUEUE_LAST                ; If yes, put at end of queue.
         24 A1   D5  04CB               TSTL    TTY$L_WB_IRP(R1)          ; Else, see if this entry has an
                     04CE                                                 ; associated IRP.
           09    12  04CE               BNEQ    INSERT_PKT                ; If yes, branch to insert
                     04D0                                                 ; packet before it.
   54      51    D0  04D0               MOVL    R1,R4                     ; Otherwise, go on to next
           EE    11  04D3               BRB     10$                       ; entry in queue.
                     04D5
                     04D5       QUEUE_LAST:                               ; Queue at end of queue.
   54    04 A0   D0  04D5               MOVL    TTY$L_WB_BLINK(R0),R4     ; Get back pointer.
                     04D9
                     04D9       INSERT_PKT:
           63    0E  04D9               INSQUE  TTY$L_WB_FLINK(R3),-      ; Insert new packet in the
           64        04DB                       TTY$L_WB_FLINK(R4)        ; queue.
           01    BA  04DC               POPR    #^M<R0>                   ; Remove saved register and
   51      6E    D0  04DE               MOVL    (SP),R1                   ; get queued address
   5E      08    C0  04E1               ADDL    #8,SP                     ; clean stack
           61    17  04E4               JMP     (R1)                      ; return to queued address
                     04E6
                     04E6       START_PKT:                                ; Start the packet.
 00D4 C5   53    D0  04E6               MOVL    R3,UCB$L_TT_WRTBUF(R5)    ; Point to packet from UCB.
                     04EB               SET_STATE -                       ; Set the write state.
                     04EB                       WRITE
```

```
                                                               I 7
TTYSTRSTP                       - Terminal driver start/stop I/O routine 16-SEP-1984 02:18:30  VAX/VMS Macro V04-00         Page  23
V04-000                         TTY$WRTSTARTIO - Starts or queues a writ  5-SEP-1984 04:17:09  [TTDRVR.SRC]TTYSTRSTP.MAR;1       (15)

                    54  D5  04EF   911          TSTL    R4                          ; If this write does not have an
                    31  13  04F1   912          BEQL    20$                         ; IRP, don't check IRP fields.
           62    40 A4  C8  04F3   913          BISL    IRP$Q_TT_STATE(R4),(R2)     ; Set write state bits.
        04 A2    44 A4  C8  04F7   914          BISL    IRP$Q_TT_STATE+4(R4),4(R2)        ; Set write state bits.
                 38 A4  D4  04FC   915          CLRL    IRP$L_MEDIA(R4)             ; Set up IRP for completion.
                    07  E1  04FF   916          BBC     #IO$V_ENABLMBX,-            ; Branch if enable-mailbox
              15 20 A4      0501   917                  IRP$W_FUNC(R4),10$          ; is not requested.
                            0504   918
           51   00C0 C5  D0  0504   919          MOVL    UCB$L_TT_LOGUCB(R5),R1      ; GET LOGICAL UCB ADDRESS
     44 A5   00010000 8F  CA  0509   920          BICL    #TT$M_MBXDSABL,UCB$L_DEVDEPEND(R5) ; CLEAR MAILBOX DISABLED PHYSICAL
     44 A1   00010000 8F  CA  0511   921          BICL    #TT$M_MBXDSABL,UCB$L_DEVDEPEND(R1) ; CLEAR MAILBOX DISABLED LOGICAL
                            0519   922
                            0519   923 10$:
                    06  E1  0519   924          BBC     #IO$V_CANCTRLO,-            ; Branch if cancel control-O
              11 20 A4      051B   925                  IRP$W_FUNC(R4),30$          ; is not requested.
                            051E   926          CLR_STATE -                         ; Clear control-O state.
                            051E   927                  CTRLO
                    0B  11  0522   928          BRB     30$                         ; And start the output.
                            0524   929
                            0524   930 ;
                            0524   931 ; Start a broadcast packet.
                            0524   932 ;
                            0524   933
                            0524   934 20$:
                 0B A3  94  0524   935          CLRB    TTY$B_WB_FIPL(R3)           ; Indicate block free to fork (for DMA)
              04 10 A3  E9  0527   936          BLBC    TTY$L_WB_FR3(R3),30$        ; Branch if ANSI broadcast or norefresh
                            052B   937          SET_STATE -
                            052B   938                  <REFRSH>                    ; Set refresh read
                            052F   939 30$:                                         ; Go output buffer.
                    03  BA  052F   940          POPR    #^M<R0,R1>                  ; Restore saved register
                            0531   941                                              ; and queued address.
                        05  0531   942          RSB                                 ; And return to caller.
```

```
TT
Sy

CL
CN
CO
DO
DO
DO
DO
DO
DO
DO
DO
DO
DO
EN
F
FI
IN
IO
IO
IO
IO
IO
IO
IO
IO
IO
IO
IO
IO
IO
IP
IR
IR
IR
IR
IR
IR
IR
IR
IR
IR
IR
IR
IR
IR
IR
IR
IR
MO
MO
NO
PO
PR
PR
QU
QU
RE
```

```
                        0532          .SBTTL  TTY$STARTOUTPUT - START OUTPUT OPERATION ON UNIT
                        0532
                        0532     ;++
                        0532     ; TTY$STARTOUTPUT - START OUTPUT ON UNIT
                        0532     ;
                        0532     ; FUNCTIONAL DESCRIPTION:
                        0532     ;
                        0532     ; THIS ROUTINE IS USED TO INITIATE OUTPUT ON A UNIT. THIS OPERATION STARTS
                        0532     ; THE FLOW OF DATA EVEN IN THE CASE OF READS. THE ACTION IS TO TEST THE
                        0532     ; STATE OF INTERRUPT EXPECTED. IF AN INTERRUPT IS EXPECTED, THEN NOTHING NEED BE DON
                        0532     ; BECAUSE A SUBSEQUENT INTERRUPT WILL CONTINUE APPROP. WITH THE CURRENT STATE.
                        0532     ; IF NO INTERRUPT IS EXPECTED, THEN THE TTY$GETNEXTCHAR ROUTINE IS ENTERED TO RETURN
                        0532     ; THE NEXT CHARACTER(S) FOR THE UNIT. THEN IF AVAILABLE THE PORT DRIVER
                        0532     ; STARTIO ROUTINE IS ENTERED.
                        0532     ; THIS OPERATION IS IDENTICAL TO THE OPERATION OF AN OUTPUT READY INTERRUPT.
                        0532     ;
                        0532     ; INPUTS:
                        0532     ;
                        0532     ;       R2 = ADDRESS OF THE UNIT STATE VECTOR
                        0532     ;       R5 = UCB ADDRESS
                        0532     ;
                        0532     ; OUTPUTS:
                        0532     ;
                        0532     ;       NONE
                        0532     ;--
                        0532     TTY$STARTOUTPUT::                                     ; START OUTPUT
10 64 A5     01     E0  0532              BBS     #UCB$V_INT,UCB$W_STS(R5),100$; LEAVE HERE IF INTERRUPT EXPECTED
 50   0118 C5       D0  0537              MOVL    UCB$L_TT_PORT(R5),R0        ; GET THE PORT'S VECTOR TABLE ADDRESS
          FAC1'     30  053C              BSBW    TTY$GETNEXTCHAR             ; GET NEXT CHARACTER FOR UNIT
    03 010B C5      E9  053F              BLBC    UCB$B_TT_OUTYPE(R5),100$;  LEAVE IF NOTHING TO OUTPUT
         00 B0      17  0544              JMP     @PORT_STARTIO(R0)           ; START OUTPUT ON LINE
                    05  0547     100$:    RSB                                 ; RETURN
```

```
                          0548          .SBTTL  TTY$GETNXTWRITE - Start next write or restart read
                          0548
                          0548  ;++
                          0548  ;
                          0548  ; Functional description:
                          0548  ;
                          0548  ;       This routine gains control at device IPL on return from the
                          0548  ;       VMS fork queuing routine. The routine tries to restart a
                          0548  ;       suspended but now active read, or to dequeue and start the
                          0548  ;       next write request vis WRTSTARTIO.
                          0548  ;
                          0548  ;       The routine always returns to the caller of TTY$WRITEDONE,
                          0548  ;       TTY$READONE, or BRDCST in TTYCHARO. This caller is usually
                          0548  ;       GETNEXTCHAR, so setting states causes the driver to go on
                          0548  ;       echoing and outputting.
                          0548  ;
                          0548  ; Inputs:
                          0548  ;
                          0548  ;       0(SP)   - address of the UCB state vector
                          0548  ;       4(SP)   - address of the UCB
                          0548  ;
                          0548  ; Outputs:
                          0548  ;
                          0548  ;       R2      - address of the UCB state vector
                          0548  ;       R3      - address of a write buffer if writing is to begin
                          0548  ;       R5      - address of the UCB
                          0548  ;
                          0548  ;       The 2 named inputs are removed from the stack.
                          0548  ;
                          0548  ;--
                          0548
                          0548  TTY$GETNXTWRITE::                               ; Check for a new write.
                 24   BA  0548          POPR    #^M<R2,R5>                      ; Restore UCB state address and
                          054A                                                  ; UCB address.
                          054A          IF_NOT_STATE -                          ; If not in a read state, just
                          054A                  READ,10$                        ; branch forward,
                          054E          IF_STATE -                              ; If noecho, don't block writes
                          054E                  NOECHO,10$
        00CC C5       D1  0552          CMPL    UCB$L_TT_WFLINK(R5),-
        00D0 C5           0556                  UCB$L_TT_WBLINK(R5)             ; queue empty?
             0A       13  0559          BEQL    5$                              ; Branch if yes
   53   00CC C5       D0  055B          MOVL    UCB$L_TT_WFLINK(R5),R3          ; Fetch address of next irp
             09       E0  0560          BBS     #IO$V_BREAKTHRU,-
          09 20 A3        0562                  IRP$W_FUNC(R3),10$              ; Start if break thru write.
     53    78 A5      D0  0565  5$:     MOVL    UCB$L_SVAPTE(R5),R3             ; get the read packet address
           3C A3      B5  0569          TSTW    TTY$W_RB_TXTOFF(R3)             ; see if any input
                          056C                                                  ; has been received.
              15      12  056C          BNEQ    20$                             ; If yes, go restart read.
                          056E
                          056E  10$:    IF_STATE WRITE,30$                      ; if we are writing then don't get the
                          0572                                                  ; next write
                          0572                                                  ; Otherwise, look for a write.
   53   00CC D5       0F  0572          REMQUE  @UCB$L_TT_WFLINK(R5),R3         ; Get a new write buffer.
             0A       1D  0577          BVS     20$                             ; Branch if no buffers exist.
    00000586'EF       9F  0579          PUSHAB  30$                             ; Save a write start return
                          057F                                                  ; address.
           FF0F       30  057F          BSBW    WRTSTARTIO                      ; Start the write.
```

```
            05  0582          RSB                          ; Return to GETNEXTCHAR.
                0583
                0583  20$:
    FA7A'   30  0583          BSBW    TTY$RESTARTIO        ; Restart the read, if any.
                0586
                0586  30$:                                 ; Joint read/write return.
            05  0586          RSB                          ; Return.
```

```
                           0587          .SBTTL  TTY$WRITEDONE - Complete a write operation
                           0587
                           0587   ;++
                           0587   ; TTY$WRITEDONE - WRITE OPERATION DONE
                           0587   ;
                           0587   ; FUNCTIONAL DESCRIPTION:
                           0587   ;
                           0587   ;       This routine creates a fork process to complete the write, and
                           0587   ;       checks for another write packet to start up.
                           0587   ;
                           0587   ; INPUTS:
                           0587   ;
                           0587   ;       R2 = ADDRESS OF THE UNIT STATE VECTOR
                           0587   ;       R5 = UCB ADDRESS
                           0587   ;
                           0587   ;       TTY$W_WB_STATUS - status of operation
                           0587   ;       TTY$W_WB_BCNT   - number of bytes transferred
                           0587   ;
                           0587   ; OUTPUTS:
                           0587   ;
                           0587   ;       R2,R5 ARE PRESERVED.
                           0587   ;
                           0587   ;--
                           0587
                           0587   TTY$WRITEDONE::                               ; Complete write operation.
                           0587
                           0587   ;
                           0587   ; This routine used to start by clearing a whole raft of state bits.
                           0587   ; I only turn off write-related bits, and I do that in TTY$GETNXTWRITE.
                           0587   ; The bits I no longer modify are:
                           0587   ;
                           0587   ;       READ, DEL, XON, EOL, PROMPT, CTRLR, NOFLTR, ESC, ESC_O, and
                           0587   ;       BADESC
                           0587   ;
                           0587
                           0587
             24      BB    0587           PUSHR   #^M<R2,R5>              ; Save state and UCB address.
                           0589   14$:
                           0589
    53    00D4 C5    D0    0589           MOVL    UCB$L_TT_WRTBUF(R5),R3  ; Get address of write buffer.
                           058E
                           058E   ;
                           058E   ; NEW LINE MODIFIER
                           058E   ;
       54    24 A3   D0    058E           MOVL    TTY$L_WB_IRP(R3),R4     ; Get address of associated IRP.
             0C      13    0592           BEQL    10$
    07 20 A4   0A    E1    0594           BBC     #IO$V_NEWLINE,IRP$W_FUNC(R4),10$; NO NEWLINE THEN DON'T ADD A THING
                           0599           SET_STATE <SENDLF,SKIPLF,NL>
                           05A0   10$:
                           05A0           CLR_STATE -                     ; Clear the write bits.
                           05A0                   <WRITE,WRTALL>
          9D AF      9F    05A8           PUSHAB  TTY$GETNXTWRITE         ; Return address after queuing fork
                           05AB
                           05AB   WRITEPOST:
                           05AB
          0B A5      90    05AB           MOVB    UCB$B_FIPL(R5),-        ; Set up fork IPL in the buffer
          0B A3            05AE                   TTY$B_WB_FIPL(R3)       ; block.
```

```
      54     24 A3    D0  05B0          MOVL    TTY$L_WB_IRP(R3),R4       ; Get address of associated IRP.
                08    12  05B4          BNEQ    5$                        ; one there then continue
                06    90  05B6          MOVB    #IPL$_QUEUEAST,-          ; for broadcast fork to ipl6
             0B A3        05B8                  TTY$B_WB_FIPL(R3)
                7E    D4  05BA          clrl    -(sp)                     ; no irp then no process to fork to
                03    11  05BC          brb     7$
                          05BE  5$:
             0C A4    DD  05BE          PUSHL   IRP$L_PID(R4)             ; FORK ON THE PID OF THE IO OWNER
         55     53    D0  05C1  7$:     MOVL    R3,R5                     ; Setup fork block address.
        00000000'EF   16  05C4          JSB     TTY$SYNCH                 ; Create a fork process.
                          05CA
                          05CA  ;
                          05CA  ; This is the write completion fork process. Registers are as follows:
                          05CA  ;
                          05CA  ;       R4      - address of IRP
                          05CA  ;       R5      - address of write buffer (TWP)
                          05CA  ;
                          05CA
         53     54    D0  05CA          MOVL    R4,R3                     ; Need IRP in R3 for I/O post.
                54    13  05CD          BEQL    100$
                          05CF
                          05CF          .IF DF  CA$_MEASURE_IOT
                          05CF  ;
                          05CF  ;ACCUMULATE STATISTICS ON NUMBER OF CHARACTERS AND I/OS TO TERMINALS.
                          05CF  ;
                          05CF          BSBB    TTSTATS                   ; CALL STATISTICS ROUTINE.
                          05CF
                          05CF          .ENDC
                          05CF
                          05CF  ;
                          05CF  ; NOTE: IRP$L_MEDIA = IRP$L_IOST1
                          05CF  ; NOTE: terminal position is 0-based; interface position is 1-based
                          05CF  ;
                          05CF
         54     55    D0  05CF          MOVL    R5,R4                     ; Put buffer address in R4.
             38 A3    3C  05D2          MOVZWL  IRP$L_MEDIA(R3),-         ; number of lines output for the
             3C A3        05D5                  IRP$L_IOST2(R3)           ; write QIO, and zero other values
             28 A4    D0  05D7          MOVL    TTY$W_WB_STATUS(R4),-     ; move status and count of bytes
             38 A3        05DA                  IRP$L_IOST1(R3)           ; transferred into IOSB
                          05DC
      55     1C A3    D0  05DC          MOVL    IRP$L_UCB(R3),R5          ; Regain LUCB address.
     55    00A0 C5    D0  05E0          MOVL    UCB$L_TL_PHYUCB(R5),R5    ; Switch to Physical context
                1D    13  05E5          BEQL    15$                       ; Disconnect has occured!
           00FC C5    B1  05E7          CMPW    UCB$W_TT_CURSOR(R5),-     ; Is cursor marker beyond the right-
             42 A5        05EB                  UCB$W_DEVBUFSIZ(R5)       ; hand edge of screen?
                09    1E  05ED          BGEQU   12$                       ; Branch if cursor has gone too far.
           00FC C5    A1  05EF          ADDW3   UCB$W_TT_CURSOR(R5),-     ; Else return cursor column position,
       3E A3    01        05F3                  #1,IRP$L_IOST2+2(R3)      ; adjusted for zero offset, in IOSB.
                05    11  05F6          BRB     14$                       ; Continue building IOSB.
             42 A5    B0  05F8  12$:    MOVW    UCB$W_DEVBUFSIZ(R5),-     ; If necessary, return cursor column
             3E A3        05FB                  IRP$L_IOST2+2(R3)         ; position in IOSB as right-hand edge.
           00FE C5    81  05FD  14$:    ADDB3   UCB$B_TT_LINE(R5),-       ; move line position into IOSB
       3F A3    01        0601                  #1,IRP$L_IOST2+3(R3)      ;
                          0604
             08 A4    B0  0604  15$:    MOVW    TTY$W_WB_SIZE(R4),-       ; Move size of buffer into IRP
             30 A3        0607                  IRP$W_BOFF(R3)            ; to record quota used.
      55     1C A3    D0  0609          MOVL    IRP$L_UCB(R3),R5          ; Restore logical UCB address
```

```
         58 A5    53   D1 060D 1156            CMPL    R3,UCB$L_IRP(R5)          ; Is this the current write
                  06   13 0611 1157            BEQL    30$                       ; blocking the i/o queue?
                          0613 1158
      00000000'GF      17 0613 1159 20$:       JMP     G^COM$POST                ; Full duplex: complete write
                          0619 1160
                          0619 1161 30$:                                         ; Half duplex:
         50    38 A3   7D 0619 1162            MOVQ    IRP$L_MEDIA(R3),R0        ; Load IOST1 and IOST2 in R0,R1
                          061D 1163            REQCOM                            ; Complete request and get next
                          0623 1164                                              ; entry in system queue.
                          0623 1165 100$:
         53    20 A5   D0 0623 1166            MOVL    TTY$L_WB_END(R5),R3       ; GET THE ADDRESS OF THE LAST CHARACTER
               2C B5   17 0627 1167            JMP     @TTY$L_WB_RETADDR(R5)     ; Want fork process to gain control
                          062A 1168                                              ; string for fork process use.
                          062A 1169
                          062A 1170
```

```
                        062A            .SBTTL  TTY$WRITEPOST - QUEUE A WRITE COMPLETION
                        062A
                        062A    ;++
                        062A    ; TTY$WRITEPOST - QUEUE A WRITE COMPLETION
                        062A    ;
                        062A    ; FUNCTIONAL DESCRIPTION:
                        062A    ;
                        062A    ;       THIS ROUTINE FORKS ON A TWP TO COMPLETE A QUEUED WRITE OPERATION
                        062A    ;       BOTH HALF AND FULL DUPLEX.
                        062A    ;
                        062A    ; INPUTS:
                        062A    ;
                        062A    ;       R4 = TWP ADDRESS
                        062A    ;       R5 = UCB ADDRESS
                        062A    ;
                        062A    ;       UCB$W_BOFF - STATUS OF OPERATION
                        062A    ;
                        062A    ; OUTPUTS:
                        062A    ;
                        062A    ;       R0,R1,R2,R3,R5 ARE PRESERVED
                        062A    ;--
                        062A
                        062A    TTY$WRITEPOST::
                        062A
            28      BB  062A            PUSHR   #^M<R3,R5>              ; SAVE REGISTERS
       53   54      D0  062C            MOVL    R4,R3                   ; TWP ADDRESS
         2A A3      B4  062F            CLRW    TTY$W_WB_BCNT(R3)       ; NONE TRANSFERED
         7C A5      B0  0632            MOVW    UCB$W_BOFF(R5),-        ; SAVE COMPLETION STATUS
         28 A3          0635                    TTY$W_WB_STATUS(R3)
          FF71      30  0637            BSBW    WRITEPOST               ; QUE THE FORK
            28      BA  063A            POPR    #^M<R3,R5>
                    05  063C            RSB
                        063D
                        063D
```

```
063D
063D            .IF DF  CA$_MEASURE_IOT
063D    ;
063D    ;       Subroutine to accumulate statistics on the number of
063D    ;       the number of characters read and written to terminals
063D    ;
063D    TTSTATS:BLBC    G^PMS$GL_DOSTATS,40$            ; IF FLAG SET, BYPASS STATISTICS CODE
063D            MOVZWL  IRP$W_BCNT(R3),R1               ; GET # CHARACTERS TRANSFERRED.
063D            DIVL3   #5,R1,R0                        ; STATISTICS ARE KEPT IN INCREMENTS
063D                                                    ;       OF 5 CHARACTERS.
063D            CMPL    #9,R0                           ; LAST ENTRY IN TABLE IS FOR I/OS
063D                                                    ;       OF >= 45 CHARACTERS.
063D            BGEQ    10$
063D            MOVL    #9,R0
063D    10$:    CMPZV   #IRP$V_FCODE,#IRP$S_FCODE,IRP$W_FUNC(R3),#TTY$C_FC_READ
063D                                                    ; CHECK IF JUST FINISHED A READ OR WRITE.
063D            BNEQ    20$                             ; BRANCH FOR WRITE
063D    ;
063D    ; COMPILE STATISTICS FOR READ
063D    ;
063D            INCL    G^PMS$AL_READTBL[R0]            ; INCREMENT APPROPRIATE RANGE.
063D            INCL    G^PMS$GL_TREADS                 ; INCREMENT READ COUNT
063D            ADDL2   R1,G^PMS$GL_READCNT             ; INCREMENT TOTAL COUNT FOR CHARACTERS
063D            BRB     40$
063D    ;
063D    ; COMPILE STATISTICS FOR WRITE
063D    ;
063D    20$:    INCL    G^PMS$AL_WRITETBL[R0]           ; INCREMENT APPROPRIATE RANGE.
063D            INCL    G^PMS$GL_TWRITES                ; INCREMENT WRITE COUNT
063D            ADDL2   R1,G^PMS$GL_WRTCNT              ; INCREMENT TOTAL COUNT FOR CHARACTERS
063D                                                    ;       WRITTEN.
063D    40$:    RSB                                     ; RETURN TO CALLER.
063D
063D            .ENDC
```

```
                         063D          .SBTTL  TTY$READONE - READ OPERATION DONE
                         063D   ;++
                         063D   ; TTY$READONE - READ I/O OPERATION DONE
                         063D   ;
                         063D   ; FUNCTIONAL DESCRIPTION:
                         063D   ;
                         063D   ; THIS ROUTINE IS ENTERED TO COMPLETE THE CURRENT READ OPERATION.
                         063D   ; THE ACTION IS TO RESET THE STATE OF THE UNIT TO REFLECT THE CHANGE AND TO
                         063D   ; FORK ON THE IRP TO COMPLETE THE PROCESSING.
                         063D   ;
                         063D   ; INPUTS:
                         063D   ;
                         063D   ;       R2 = ADDRESS OF THE UNIT STATE VECTOR
                         063D   ;       R5 = UCB ADDRESS
                         063D   ;
                         063D   ;               UCB$W_BOFF = STATUS WORD
                         063D   ;               UCB$W_BCNT = COUNT OF TRANSFER
                         063D   ;
                         063D   ;       IRP$L_MEDIA(CURRENT PACKET) = TERMINATOR AND TERMINATOR SIZE
                         063D   ;
                         063D   ; OUTPUTS:
                         063D   ;
                         063D   ;
                         063D   ;       NONE
                         063D   ;--
                         063D   .enable lsb
                         063D   TTY$READONE::                                   ; READ I/O DONE
     03 44 A5    12   E1 063D           BBC     #TT$V_READSYNC,UCB$L_DEVDEPEND(R5),10$; BR IF NOT READSYNC
           F9BB'  30 0642           BSBW    TTY$XOFF                        ; SEND XOFF
        68 A5    02   AA 0645   10$:    BICW    #UCB$M_TT_TIMO,UCB$W_DEVSTS(R5); CLEAR TIMEOUT ENABLED
                         0649   ;
                         0649   ; SET UP ERRORS ON ESCAPE SEQUENCES
                         0649   ;
                         0649           IF_NOT_STATE ESC,15$            ; IF NOT ESCAPE THEN BR
  7C A5     01FC 8F   B0 064D           MOVW    #SS$_PARTESCAPE,UCB$W_BOFF(R5); ASSUME PARTIAL ESCAPE SEQUENCE
     54       78 A5   D0 0653           MOVL    UCB$L_SVAPTE(R5),R4     ; GET THE ADDRESS OF THE READ PACKET
     53       58 A5   D0 0657           MOVL    UCB$L_IRP(R5),R3        ; ADDRESS CURRENT PACKET
     53       3A A3   9B 065B           MOVZBW  IRP$L_MEDIA+2(R3),R3    ; MAKE THE ESCAPE SEQUENCE COUNT A WORD
     3C A4       53   A2 065F           SUBW    R3,TTY$W_RB_TXTOFF(R4)  ; SUBTRACT OUT TERMINATOR LENGTH
                         0663   15$:    IF_NOT_STATE BADESC,20$         ; ESCAPE SYNTAX CORRECT?
        7C A5    3C   B0 0667           MOVW    #SS$_BADESCAPE,UCB$W_BOFF(R5); SET STATUS FOR IMPROPER ESCAPE SEQ
                         066B   ;
                         066B   ; RESET PASSALL AND NOECHO IF MODES
                         066B   ;
                         066B   20$:
                         066B
                         066B           .IF DF  CA$_MEASURE_IOT
                         066B
                         066B           BLBC    G^PMS$GL_DOSTATS,25$    ; IF FLAG SET, BYPASS STATISTICS CODE
                         066B           BBC     #TT$V_PASSALL,UCB$L_DEVDEPEND(R5),25$; BR IF NOT PASSALL
                         066B           INCL    G^PMS$GL_PASSALL        ; INCREMENT PASSALL COUNT
                         066B
                         066B           .ENDC
                         066B
  54   44 A5    FC 8F   8B 066B   25$:    BICB3   #^C<TT$M_PASSALL!TT$M_NOECHO>,UCB$L_DEVDEPEND(R5),R4;
04 A2   02      02  54   F0 0671           INSV    R4,#TTY$V_ST_PASALL,#2,4(R2)            ; RESET PASSALL AND NOECHO
54     44 A5    01  03   EE 0677           EXTV    #TT$V_ESCAPE,#1,UCB$L_DEVDEPEND(R5),R4  ; GET CURRENT SETTING
```

```
 04 A2   01   0B   54   F0 067D          INSV    R4,#TTY$V_ST_ESCAPE,#1,4(R2)             ; UPDATE IN STATE
    04 48 A5   12   E1 0683          BBC     #TT2$V_PASTHRU,UCB$L_DEVDEPND2(R5),98$  ; IN PASS THRU MODE
                       0688          SET_STATE PASALL
                       068C      98$:
                       068C
                       068C          CLR_STATE -                              ; Clear state bits relevant to
                       068C                  <READ,DEL,EOL,-                  ; a read (this does not include
                       068C                  PROMPT,CTRLR,NOFLTR,-            ; the write bits).
                       068C                  ESC,ESC_O,BADESC,PRE,TERMNORM,-
                       068C                  REFRSH,EDITREAD,SKIPCRLF,RDVERIFY,ECHAES,-
                       068C                  MULTI,RECALL,OVERSTRIKE,EDITING,QUOTING,BACKSPACE>
         53    58 A5   D0 0699          MOVL    UCB$L_IRP(R5),R3                 ; ADDRESS CURRENT PACKET
         54    78 A5   D0 069D          MOVL    UCB$L_SVAPTE(R5),R4              ; GET THE ADDRESS OF THE READ PACKET
    012B C5    3A A4   90 06A1          MOVB    TTY$W_RB_CPZORG(R4),UCB$B_TT_OLDCPZORG(R5); SAVE BECAUSE
                          06A7                                                   ; WE MAY CONTINUE THIS READ
      3C A3    0C A3   D0 06A7          MOVL    IRP$L_PID(R3),IRP$L_TT_TERM(R3); SAVE DATA FOR FORK BLOCK
      4C A3    0B A3   90 06AC          MOVB    IRP$B_RMOD(R3),IRP$Q_TT_PRMPT(R3);
      0B A3    0B A5   90 06B1          MOVB    UCB$B_FIPL(R5),IRP$B_RMOD(R3); SET FORK IPL
      30 A3    7C A5   B0 06B6          MOVW    UCB$W_BOFF(R5),IRP$W_BOFF(R3); SAVE STATUS
      32 A3    3C A4   B0 06BB          MOVW    TTY$W_RB_TXTOFF(R4),IRP$W_BCNT(R3); SAVE TRANSFER SIZE
                  24   BB 06C0          PUSHR   #^M<R2,R5>                       ;
           55     53   D0 06C2          MOVL    R3,R5                            ; SET UP FORK BLOCK ADDRESS
         53    10 A3   7D 06C5          MOVQ    IRP$L_AST(R3),R3                 ; GET PARAMS FOR FORK
             FE7B CF   9F 06C9          PUSHAB  W^TTY$GETNXTWRITE                ; FORCE RETURN TO RESTORE UCB ADDRESS
                3C A5  DD 06CD          PUSHL   IRP$L_TT_TERM(R5)                ; USE THE PID OF THE ISSUING PROCESS
        00000000'EF    16 06D0          JSB     TTY$SYNCH                        ; CREATE FORK PROCESS
                          06D6      ;
                          06D6      ; I/O DONE FORK PROCESS
                          06D6      ;
         53    55      D0 06D6          MOVL    R5,R3                            ; RESTORE PACKET AND UCB ADDRESSES
                          06D9
                          06D9          .IF DF  CA$_MEASURE_IOT
                          06D9      ;
                          06D9      ;ACCUMULATE STATISTICS ON NUMBER OF CHARACTERS AND I/OS TO TERMINALS.
                          06D9      ;
                          06D9          BSBW    TTSTATS                          ; CALL STATISTICS ROUTINE.
                          06D9
                          06D9          .ENDC
                          06D9
         55    1C A3   D0 06D9          MOVL    IRP$L_UCB(R3),R5                 ; GET LUCB ADDRESS FROM IRP
       55    00A0 C5   D0 06DD          MOVL    UCB$L_TL_PHYUCB(R5),R5           ; SWITCH TO PHYSICAL USB CONTEXT
         50    30 A3   D0 06E2          MOVL    IRP$W_BOFF(R3),R0                ; GET STATUS AND TRANSFER SIZE
         54    3A A3   9B 06E6          MOVZBW  IRP$L_MEDIA+2(R3),R4             ; GET THE BYTE COUNT OF THE TERMINATOR
      32 A3       54   A0 06EA          ADDW    R4,IRP$W_BCNT(R3)                ; CALC TOTAL TRANSFER SIZE
         54    2C A3   D0 06EE          MOVL    IRP$L_SVAPTE(R3),R4              ; GET ADDRESS OF THE BUFFER BLOCK
   21 20 A3       0F   E1 06F2          BBC     #IO$V_EXTEND,IRP$W_FUNC(R3),27$
      39 A3    FF 8F   90 06F7          MOVB    #^XOFF,IRP$L_MEDIA+1(R3); INFORCE RESERVED FIELD
   17 40 A3       0A   E0 06FC          BBS     #TTY$V_ST_RDVERIFY,IRP$Q_TT_STATE(R3),27$; DON'T DO IT FOR READ VERI
         51    30 A4   3C 0701          MOVZWL  TTY$W_RB_CINOFF(R4),R1           ; get the offset to this character
      2C A4       51   C0 0705          ADDL    R1,TTY$L_RB_LIN(R4)              ; ADD IN THE OFFSET TO GET AN ADDRESS
         51    3C A4   3C 0709          MOVZWL  TTY$W_RB_TXTOFF(R4),R1           ; GET THE LENGTH OF THE READ
           51     64   C0 070D          ADDL    TTY$L_RB_TXT(R4),R1              ; AND GET THE ADDRESS OF THE LAST CHARACTER
         51    2C A4   C2 0710          SUBL    TTY$L_RB_LIN(R4),R1              ; SUBTRACT OUT THE ADDRESS OF THE
      3B A3       51   90 0714          MOVB    R1,IRP$L_MEDIA+3(R3)             ; LAST CHARACTER AND THIS IS THE
                          0718                                                   ; OFFSET FROM THE END OF THE READ
                          0718                                                   ; TO THE LAST POSITION.
         51    38 A3   D0 0718      27$:    MOVL    IRP$L_MEDIA(R3),R1           ; SET TERMINATOR DATA
         0C A3    3C A3    D0  071C               MOVL    IRP$L_TT_TERM(R3),IRP$L_PID(R3); RETURN IRP DATA
         0B A3    4C A3    90  0721               MOVB    IRP$W_TT_PRMPT(R3),IRP$B_RMOD(R3);
         30 A3    08 A4    B0  0726               MOVW    TTY$W_RB_SIZE(R4),IRP$W_BOFF(R3); MAKE IT QUOTA
         0F 48 A5    0C    E0  072B               BBS     #TT2$V_EDITING,UCB$L_DEVDEPND2(R5),40$; IF EDITING THEN
                               0730                                               ; SAVE THE BUFFER.
                               0730       READ$DONE:
         4F 48 A5    0E    E0  0730               BBS     #TT2$V_FALLBACK,UCB$L_DEVDEPND2(R5),200$; DO WE HAVE TO CHECK
                               0735       END_FALL:                               ; FOR INPUT FALLBACK
                               0735
                               0735       TTY$DONE:
            55    1C A3    D0  0735               MOVL    IRP$L_UCB(R3),R5         ; RESTORE LOGICAL UCB ADDRESS
                               0739               REQCOM                           ; COMPLETE REQUEST
                               073F       ;
                               073F       ; SAVE THE COMMAND IF WE ARE IN EDITING, THE READ WAS SUCESSFUL AND
                               073F       ; IT WAS NOT A NOECHO READ.
                               073F       ;
          022C 8F       50     B1  073F  40$:     CMPW    R0,#SS$_TIMEOUT          ; ALLOW TIMEOUT ERRORS TO BE SAVED
                        03     13  0744           BEQL    42$                      ; ...
                     E7 50     E9  0746           BLBC    R0,READ$DONE             ; READ NOT SUCCESSFUL THEN DON'T SAVE
            44 A5       03     D3  0749  42$:     BITL    #TT$M_NOECHO!TT$M_PASSALL,UCB$L_DEVDEPEND(R5);DON'T COPY ON
                        E1     12  074D           BNEQ    READ$DONE                ; PASSALL OR NOECHO
          0000004C      8F     D3  074F           BITL    #TTY$M_ST_NOECHO!TTY$M_ST_PASALL!TTY$M_ST_NOFLTR,-
                     44 A3         0755                   IRP$Q_TT_STATE+4(R3)     ; WERE WE PASS ALL, NOECHO
                        D7     12  0757           BNEQ    READ$DONE                ; OR NO FILTER THEN DON'T SAVE THIS BUFFER
                        3F     BB  0759           PUSHR   #^M<R0,R1,R2,R3,R4,R5>   ; SAVE THE REGISTERS OVER THE MOVE
            50    3C A4     3C     075B           MOVZWL  TTY$W_RB_TXTOFF(R4),R0   ; GET THE LENGTH
                        1F     13  075F           BEQL    35$                      ; IF THE LENGTH IS ZERO THEN DON'T SAVE
                                   0761                                            ; THE DATA
     50    00000100 8F     D1  0761           CMPL    #TTY$K_TA_RCLLEN,R0      ; DOES ALL THE DATA FIT?
                        07     18  0768           BGEQ    30$                      ; YES THEN CONTINUE ON
     50    00000100 8F     D0  076A           MOVL    #TTY$K_TA_RCLLEN,R0      ; ELSE USE THE WHOLE BFFER
         51     00E4 C5     D0  0771  30$:     MOVL    UCB$L_TT_TYPAHD(R5),R1   ; GET THE ADDRESS OF THE TYPEAHEAD BUFFER
            14 A1       50     B0  0776           MOVW    R0,TTY$W_TA_RCLSIZ(R1)   ; KEEP THE SIZE CORRECTLY
   18 A1    00 B4       50     28  077A           MOVC3   R0,@TTY$L_RB_TXT(R4),TTY$A_TA_RCL(R1); MOVE THE DATA IN
                        3F     BA  0780  35$:     POPR    #^M<R0,R1,R2,R3,R4,R5>   ; RESTORE THE REGISTERS AND
                        AC     11  0782  91$:     BRB     READ$DONE
                                   0784
                                   0784       ;
                                   0784       ; INPUT FALLBACK TABLE IMPLIMENTATION
                                   0784       ;
                                   0784       200$:
                        3F     BB  0784           PUSHR   #^M<R0,R1,R2,R3,R4,R5>   ; SAVE THE REGISTERS OVER THE MOVE
     51    00000000'FF     D0  0786           MOVL    @TTY$A_INPFALL,R1        ; ANY INPUT FALLBACK
                        0F     13  078D           BEQL    210$
            50    3C A4     3C     078F           MOVZWL  TTY$W_RB_TXTOFF(R4),R0   ; GET THE LENGTH
                        09     13  0793           BEQL    210$                     ; IF THE LENGTH IS ZERO THEN DON'T SAVE
   61    00     00 B4       50     2E  0795           MOVTC   R0,@TTY$L_RB_TXT(R4),#0,(R1),-
                00 B4       50         079B                   R0,@TTY$L_RB_TXT(R4)     ; MOVE THE DATA THRU THE FALLBACK TABLE
                        3F     BA  079E  210$:    POPR    #^M<R0,R1,R2,R3,R4,R5>   ; RESTORE THE REGISTERS AND
                      FF92     31  07A0  220$:    BRW     END_FALL
                                   07A3       .Disable lsb
```

```
07A3
07A3          .SBTTL  End of module
07A3
07A3          .END
```

```
                                                     I  8
TTYSTRSTP                 - Terminal driver start/stop I/O routine 16-SEP-1984 02:18:30  VAX/VMS Macro V04-00        Page  36
Symbol table                                                        5-SEP-1984 04:17:09  [TTDRVR.SRC]TTYSTRSTP.MAR;1        (24)

CLASS_MODEM_DIS     ********   X  02    SS$_ABORT             = 0000002C
CNT               = 00000001            SS$_BADESCAPE         = 0000003C
COM$POST            ********   X  02    SS$_NOPRIV            = 00000024
DO_CONNECT          00000037 R    02    SS$_NORMAL            = 00000001
DO_DISCONNECT       000000A4 R    02    SS$_NOSUCHDEV         = 00000908
DO_EXIT             00000422 R    02    SS$_PARTESCAPE        = 000001FC
DO_HANGUP           000000B5 R    02    SS$_TIMEOUT           = 0000022C
DO_MAINT            000000BF R    02    START_PKT               000004E6 R    02
DO_READ             0000010D R    02    T                     = 00000005
DO_SET              000001C1 R    02    TRANSITION_NOCHECK      ********   X  02
DO_SETC             000001B5 R    02    TT$C_BAUD_9600        = 0000000F
DO_SETM             000001B2 R    02    TT$M_DS_DTR           = 00000002
DO_WRITE            00000444 R    02    TT$M_DS_RTS           = 00000010
END_FALL            00000735 R    02    TT$M_DS_SECTX         = 00000008
F                 = 00000000            TT$M_MBXDSABL         = 00010000
FIND_BOL_NOCLEAR    ********   X  02    TT$M_MODEM            = 00200000
INSERT_PKT          000004D9 R    02    TT$M_NOBRDCST         = 00020000
IO$M_FCODE        = 0000003F            TT$M_NOECHO           = 00000002
IO$V_BREAKTHRU    = 00000009            TT$M_ODD              = 00000080
IO$V_CANCTRLO     = 00000006            TT$M_PARITY           = 00000040
IO$V_ENABLMBX     = 00000007            TT$M_PASSALL          = 00000001
IO$V_EXTEND       = 0000000F            TT$M_REMOTE           = 00002000
IO$V_LOOP         = 00000007            TT$V_ALTDISPAR        = 0000000A
IO$V_LOOP_EXT     = 0000000C            TT$V_ALTFRAME         = 00000004
IO$V_NEWLINE      = 0000000A            TT$V_ALTRPAR          = 00000005
IO$V_PURGE        = 0000000B            TT$V_CRFILL           = 0000000A
IO$V_SET_MODEM    = 0000000A            TT$V_DISPARERR        = 00000009
IO$V_TT_DISCON    = 0000000C            TT$V_EIGHTBIT         = 0000000F
IOC$REQCOM          ********   X  02    TT$V_ESCAPE           = 00000003
IPL$_QUEUEAST     = 00000006            TT$V_LFFILL           = 0000000B
IRP$B_RMOD        = 0000000B            TT$V_MODEM            = 00000015
IRP$L_ARB         = 00000058            TT$V_PARITY           = 00000006
IRP$L_AST         = 00000010            TT$V_PASSALL          = 00000000
IRP$L_IOST1       = 00000038            TT$V_READSYNC         = 00000012
IRP$L_IOST2       = 0000003C            TT$V_REMOTE           = 0000000D
IRP$L_MEDIA       = 00000038            TT$V_TTSYNC           = 00000005
IRP$L_PID         = 0000000C            TT$V_TWOSTOP          = 00000008
IRP$L_SVAPTE      = 0000002C            TT2$M_ALTYPEAHD       = 00000080
IRP$L_TT_TERM     = 0000003C            TT2$M_DCL_MAILBX      = 00000200
IRP$L_UCB         = 0000001C            TT2$M_DISCONNECT      = 00020000
IRP$L_VAL5        = 0000009C            TT2$M_DMA             = 00000040
IRP$Q_TT_STATE    = 00000040            TT2$M_XON             = 00000020
IRP$S_FCODE       = 00000006            TT2$V_ALTYPEAHD       = 00000007
IRP$V_FCODE       = 00000000            TT2$V_AUTOBAUD        = 00000001
IRP$W_BCNT        = 00000032            TT2$V_DISCONNECT      = 00000011
IRP$W_BOFF        = 00000030            TT2$V_DMA             = 00000006
IRP$W_FUNC        = 00000020            TT2$V_EDITING         = 0000000C
IRP$W_TT_PRMPT    = 0000004C            TT2$V_FALLBACK        = 0000000E
MODEM$C_INIT      = 00000000            TT2$V_HANGUP          = 00000002
MODEM$C_SHUTDWN   = 00000001            TT2$V_LOCALECHO       = 00000000
NOPRIV_EXIT         0000043C R    02    TT2$V_MODHANGUP       = 00000003
PORT_STARTIO      = 00000000            TT2$V_PASTHRU         = 00000012
PRV$V_LOG_IO      = 00000007            TT2$V_SECURE          = 00000010
PRV$V_PHY_IO      = 00000016            TT2$V_SETSPEED        = 00000008
QUEUE_LAST          000004D5 R    02    TT2$V_XON             = 00000005
QUEUE_PKT           000004BC R    02    TTY$A_INPFALL           ********   X  02
READ$DONE           00000730 R    02    TTY$A_TA_RCL          = 00000018
```

```
                                                            J  8
TTYSTRSTP                     - Terminal driver start/stop I/O routine 16-SEP-1984 02:18:30  VAX/VMS Macro V04-00        Page  37
Symbol table                                                            5-SEP-1984 04:17:09  [TTDRVR.SRC]TTYSTRSTP.MAR;1        (24)

TTY$B_WB_FIPL           = 0000000B               TTY$RESUME                ********     X   02
TTY$CRE_FORK              ********     X   02    TTY$SETUP_READ            ********     X   02
TTY$C_CR                = 0000000D               TTY$SET_LINE              ********     X   02
TTY$C_FC_SETC           = 00000003               TTY$STARTIO               00000000 RG      02
TTY$DONE                  00000735 R       02    TTY$STARTOUTPUT           00000532 RG      02
TTY$DS_SET                ********     X   02    TTY$SYNCH                 ********     X   02
TTY$GETNEXTCHAR           ********     X   02    TTY$V_FD_DISCONNECT     = 00000002
TTY$GETNXTWRITE           00000548 RG      02    TTY$V_FD_GETAHD         = 00000001
TTY$K_ER_ECHLINE        = 00000002               TTY$V_PC_DMAAVL         = 00000002
TTY$K_TA_RCLLEN         = 00000100               TTY$V_PC_NOCRLF         = 00000007
TTY$LOCK                  ********     X   02    TTY$V_ST_ESCAPE         = 0000000B
TTY$L_RB_LIN            = 0000002C               TTY$V_ST_PASALL         = 00000002
TTY$L_RB_TXT            = 00000000               TTY$V_ST_RDVERIFY       = 0000000A
TTY$L_WB_BLINK          = 00000004               TTY$V_SX_BACKSPACE      = 00000005
TTY$L_WB_END            = 00000020               TTY$V_SX_BADESC         = 00000028
TTY$L_WB_FLINK          = 00000000               TTY$V_SX_CTRLO          = 00000020
TTY$L_WB_FR3            = 00000010               TTY$V_SX_CTRLR          = 00000032
TTY$L_WB_IRP            = 00000024               TTY$V_SX_DEL            = 00000021
TTY$L_WB_RETADDR        = 0000002C               TTY$V_SX_ECHAES         = 00000039
TTY$MAINT                 ********     X   02    TTY$V_SX_EDITING        = 00000034
TTY$M_PC_DMAENA         = 00000002               TTY$V_SX_EDITREAD       = 00000009
TTY$M_PC_XOFENA         = 00000040               TTY$V_SX_EOL            = 00000008
TTY$M_ST_BACKSPACE      = 00000020               TTY$V_SX_ESC            = 00000027
TTY$M_ST_BADESC         = 00000100               TTY$V_SX_ESC_O          = 0000002E
TTY$M_ST_CTRLO          = 00000001               TTY$V_SX_MULTI          = 00000006
TTY$M_ST_CTRLR          = 00040000               TTY$V_SX_NL             = 00000029
TTY$M_ST_DEL            = 00000002               TTY$V_SX_NOECHO         = 00000023
TTY$M_ST_ECHAES         = 02000000               TTY$V_SX_NOFLTR         = 00000026
TTY$M_ST_EDITING        = 00100000               TTY$V_SX_OVERSTRIKE     = 00000037
TTY$M_ST_EDITREAD       = 00000200               TTY$V_SX_PASALL         = 00000022
TTY$M_ST_EOL            = 00000100               TTY$V_SX_PRE            = 0000003A
TTY$M_ST_ESC            = 00000080               TTY$V_SX_PROMPT         = 00000025
TTY$M_ST_ESC_O          = 00004000               TTY$V_SX_QUOTING        = 00000036
TTY$M_ST_MULTI          = 00000040               TTY$V_SX_RDVERIFY       = 0000000A
TTY$M_ST_NL             = 00000200               TTY$V_SX_READ           = 0000000C
TTY$M_ST_NOECHO         = 00000008               TTY$V_SX_RECALL         = 0000000B
TTY$M_ST_NOFLTR         = 00000040               TTY$V_SX_RECONNECT      = 0000003C
TTY$M_ST_OVERSTRIKE     = 00800000               TTY$V_SX_REFRSH         = 0000002A
TTY$M_ST_PASALL         = 00000004               TTY$V_SX_SENDLF         = 00000004
TTY$M_ST_PRE            = 04000000               TTY$V_SX_SKIPCRLF       = 00000033
TTY$M_ST_PROMPT         = 00000020               TTY$V_SX_SKIPLF         = 0000002D
TTY$M_ST_QUOTING        = 00400000               TTY$V_SX_TERMNORM       = 00000038
TTY$M_ST_RDVERIFY       = 00000400               TTY$V_SX_WRAP           = 0000002F
TTY$M_ST_READ           = 00001000               TTY$V_SX_WRITE          = 00000007
TTY$M_ST_RECALL         = 00000800               TTY$V_SX_WRTALL         = 00000024
TTY$M_ST_RECONNECT      = 10000000               TTY$WRITEDONE             00000587 RG      02
TTY$M_ST_REFRSH         = 00000400               TTY$WRITEPOST             0000062A RG      02
TTY$M_ST_SENDLF         = 00000010               TTY$WRTSTARTIO            00000452 RG      02
TTY$M_ST_SKIPCRLF       = 00080000               TTY$W_RB_CPZORG         = 0000003A
TTY$M_ST_SKIPLF         = 00002000               TTY$W_RB_LINOFF         = 00000030
TTY$M_ST_TERMNORM       = 01000000               TTY$W_RB_LINREST        = 00000032
TTY$M_ST_WRAP           = 00008000               TTY$W_RB_MODE           = 00000044
TTY$M_ST_WRITE          = 00000080               TTY$W_RB_SIZE           = 00000008
TTY$M_ST_WRTALL         = 00000010               TTY$W_RB_TXTOFF         = 0000003C
TTY$PURGE_AHEAD           ********     X   02    TTY$W_TA_RCLSIZ         = 00000014
TTY$READONE               0000063D RG      02    TTY$W_WB_BCNT           = 0000002A
TTY$RESTARTIO             ********     X   02    TTY$W_WB_SIZE           = 00000008
```

```
TTY$W_WB_STATUS                  = 00000028                       X1                                      = 00000002
TTY$XOFF                           ********    X    02            Z0                                      = 00000000
UCB$B_DEVCLASS                   = 00000040                       Z1                                      = 00000000
UCB$B_DEVTYPE                    = 00000041
UCB$B_FIPL                       = 0000000B
UCB$B_TT_CRFILL                  = 000000F6
UCB$B_TT_DEPARI                  = 000000EC
UCB$B_TT_DETYPE                  = 000000F0
UCB$B_TT_LASTC                   = 000000FF
UCB$B_TT_LFFILL                  = 000000F7
UCB$B_TT_LINE                    = 000000FE
UCB$B_TT_MAINT                   = 0000012A
UCB$B_TT_OLDCPZORG               = 0000012B
UCB$B_TT_OUTYPE                  = 0000010B
UCB$B_TT_PARITY                  = 000000F8
UCB$L_DEVDEPEND                  = 00000044
UCB$L_DEVDEPND2                  = 00000048
UCB$L_IRP                        = 00000058
UCB$L_PDT                        = 00000084
UCB$L_SVAPTE                     = 00000078
UCB$L_TL_PHYUCB                  = 000000A0
UCB$L_TT_DECHA1                  = 000000C8
UCB$L_TT_DECHAR                  = 000000C4
UCB$L_TT_LOGUCB                  = 000000C0
UCB$L_TT_PORT                    = 00000118
UCB$L_TT_TYPAHD                  = 000000E4
UCB$L_TT_WBLINK                  = 000000D0
UCB$L_TT_WFLINK                  = 000000CC
UCB$L_TT_WRTBUF                  = 000000D4
UCB$M_CANCEL                     = 00000008
UCB$M_TIMOUT                     = 00000040
UCB$M_TT_DISPARERR               = 00000002
UCB$M_TT_DSBL                    = 00000080
UCB$M_TT_LEN                     = 00000018
UCB$M_TT_NOTIF                   = 00000004
UCB$M_TT_ODD                     = 00000080
UCB$M_TT_PARTY                   = 00000040
UCB$M_TT_STOP                    = 00000020
UCB$M_TT_TIMO                    = 00000002
UCB$V_INT                        = 00000001
UCB$V_TT_DISPARERR               = 00000001
UCB$V_TT_LEN                     = 00000003
UCB$V_TT_USERFRAME               = 00000002
UCB$W_BOFF                       = 0000007C
UCB$W_DEVBUFSIZ                  = 00000042
UCB$W_DEVSTS                     = 00000068
UCB$W_STS                        = 00000064
UCB$W_TT_CURSOR                  = 000000FC
UCB$W_TT_DESPEE                  = 000000E8
UCB$W_TT_PRTCTL                  = 00000122
UCB$W_TT_SPEED                   = 000000F4
W0                               = 00001F60
W1                               = 07DC45E2
WRITEPOST                          000005AB R       02
WRTSTARTIO                         00000491 R       02
X                                = 000007DC
X0                               = 00000000
```

```
                                        +----------------+
                                        ! Psect synopsis !
                                        +----------------+

PSECT name                     Allocation          PSECT No.  Attributes
----------                     ----------          ---------  ----------
.  ABS  .                      00000000  (     0.)  00 (  0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                          00000000  (     0.)  01 (  1.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$115_DRIVER                  000007A3  (  1955.)  02 (  2.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC LONG

                                     +------------------------+
                                     ! Performance indicators !
                                     +------------------------+

Phase                    Page faults    CPU Time       Elapsed Time
-----                    -----------    --------       ------------
Initialization                   35    00:00:00.03    00:00:00.85
Command processing              118    00:00:00.33    00:00:03.16
Pass 1                          611    00:00:18.41    00:01:04.16
Symbol table sort                 0    00:00:02.66    00:00:10.55
Pass 2                          264    00:00:03.99    00:00:13.95
Symbol table output              34    00:00:00.19    00:00:00.19
Psect synopsis output             2    00:00:00.02    00:00:00.67
Cross-reference output            0    00:00:00.00    00:00:00.00
Assembler run totals           1066    00:00:25.65    00:01:33.54

The working set limit was 2100 pages.
152906 bytes (299 pages) of virtual memory were used to buffer the intermediate code.
There were 130 pages of symbol table space allocated to hold 2384 non-local and 89 local symbols.
1411 source lines were read in Pass 1, producing 18 object records in Pass 2.
56 pages of virtual memory were used to define 53 macros.

                                    +--------------------------+
                                    ! Macro library statistics !
                                    +--------------------------+

Macro library name                           Macros defined
------------------                           --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                          22
_$255$DUA28:[SYSLIB]STARLET.MLB;2                       10
TOTALS (all libraries)                                  32

2769 GETS were required to define 32 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:TTYSTRSTP/OBJ=OBJ$:TTYSTRSTP MSRC$:TTYSTRSTP/UPDATE=(ENH$:TTYSTRSTP)+EXECML$/LIB
```

0404 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

TTYSYNCH
LIS

TTYSTRSTP
LIS

TTYSUB
LIS

YCDRIVER
LIS